



## \*\* DE 6502 KENNERS \*\* -- EEN CLUB VOOR 6XXXX GEBRUIKERS

De vereniging heeft leden in Nederland, België, Duitsland, Frankrijk, Spanje, Portugal, Amerika, Zambia, Denemarken. Het doel van de vereniging is het bevorderen van de kennisuitwisseling tussen gebruikers van 6XXXX-computers, als COMMODORE-64, APPLE, CHE-1, PEARCOM, AIM-65, SYM, PET, BBC ATARI, VIC-20, BASIS 108, PROTON-computers, ITT-2020, OSI, ACC 8000, ACORN ELECTRON, SYSTEM 65, PC-100, PALLAS, MINTA FORMOSA, ORIC-1, STARLIGHT, CV-777, ESTATE III, SBC65/68, KIM, NCS, KEMPAC SYSTEM-4, Elektuur-computers (JUNIOR, en de OCTOPUS), LASER, maar ook 6800, 6809 en 68000-computers. De kennisuitwisseling wordt o.a. gerealiseerd door 6 maal per jaar DE 6502 KENNER te publiceren, door het houden van landelijke clubbijeenkomsten, door het instandhouden van een cassette-bibliotheek en door het verlenen van paperware-service. Regionale bijeenkomsten worden door de leden georganiseerd.

### Verrijningsdata DE 6502 KENNER 1985

derde zaterdag van  
februari, april, juni,  
augustus, oktober, december.

### Inlichtingen over de regio- bijeenkomsten:

Berard van Roekel  
Van der Palastraat 11 - C  
3135 LK Vlaardingen  
Tel.: 010 - 351101

De vereniging is volledig onafhankelijk, is statutair opgericht en ingeschreven bij de Kamer van Koophandel en Fabrieken voor Hollands Noorderkwartier te Alkmaar, onder nummer 634305.

Voorzitter:  
Rinus Vleesch-Dubois  
Fl. Nightingalestraat 212  
2037 NS Haarlem  
Tel.: 023 - 330993

Penningmeester:  
John F. van Sprang  
Tulp 71  
2925 EW Kripen/IJssel.  
Tel.: 01807 - 20589

Leden:  
Adri Hankel (05490 - 51151) Hardware/software/DOS65  
Erwin Visschedijk (05490 - 71416) Hardware/software/DOS65  
Gert van Opbroek (01729 - 8636)  
Nico de Vries (010 - 502239) Hardware/software/PET  
Erevoorzitter: Siep de Vries  
Ereleden: Mm. H. de Vries - Van der Winden  
Anton Müller  
Lidmaatschap: Hfl. 45,- per kalenderjaar, postrekening 3757649 t.n.v. Penningmeester KIM Gebruikers Club Ned., Kripen/IJssel.

Advertenties: Tarieven op aanvraag bij de redactie.

### Bijeenkomsten van de club

derde zaterdag van  
januari, maart, mei,  
september, november.

### Redactie-adres en informatie over paperware etc.

Willea L. van Pelt  
Jacob Jordaensstraat 15  
2923 CK Kripen/IJssel.  
Tel.: 01807 - 19881

## \*\* DE 6502 KENNER \*\* -- EEN BLAD VOOR 6XXXX GEBRUIKERS

DE 6502 KENNER is een uitgave van de KIM Gebruikers Club Nederland. Het blad wordt verstrekt aan leden van de club. DE 6502 KENNER wordt van kopij voorzien door leden van de club, bij de opmaak van een publikatie bijgestaan door de redactie. De inzendingen van programma's dienen voorzien te zijn van commentaar in de listings en zo mogelijk door een inleiding voorafgegaan. Publikatie van een inzending betekent niet dat de redactie of het bestuur enige aansprakelijkheid aanvaardt voor de toepassing ervan. De inzendingen kunnen geschieden in assembly-source-listings, in Basic, in Basicode, Forth, Focal, Comal, Pascal, Fortran, Cobol, Logo Elan, etc. etc.

De leden schrijven ook artikelen over de door hen ontwikkelde hardware en/of aanpassingen daarop. Zij schrijven tevens artikelen van algemene aard of reageren op publikaties van andere inzenders.

### DE 6502 KENNER IS EEN BLAD VAN EN DOOR DE LEDEN

Micro-ADE Assembler/Disassembler/Editor is een produkt van Micro Ware Ltd., geschreven door Peter Jennings en bestemd voor alle 6502-computers. De KIM Gebruikers Club Ned. heeft de copyrights verworven nadat ons lid Sebo Woldringh de 4 K KIM-1 versie uitbreidde tot 8 K KIM-1 versie. Adri Hankel paste deze aan voor de JUNIOR. Willea L. van Pelt stelde een nieuwe 8 K source-listing voor de JUNIOR samen.

De implementatie op andere systemen dan de KIM-1 en JUNIOR kan eenvoudig gebeuren door het aanpassen van de I/O-adressen, welke in de source-listing gemakkelijk te vinden zijn.

FATE Formaat-listener/cond. Assembler/Tape-utilities/Editor is de door ons lid Rob Banen geschreven source-listing van een 12 K universeel systeem voor de JUNIOR-computer aan de hand van het universele disk operating systeem van de fa. Proton Electronics te Naarden, nu geschikt voor werken met tapes. FATE wordt beschikbaar gesteld met toestemming van Proton.

In de edities van DE 6502 KENNER worden regelmatig mededelingen gedaan over de door de club georganiseerde bijeenkomsten. Ook worden bestuurlijke mededelingen gedaan, naast informatie over hetgeen in de handel te koop is. Leden die iets te koop hebben of iets zoeken kunnen dit in de edities van DE 6502 KENNER bekend maken. Ook worden brieven aan de redactie gepubliceerd, evenals specifieke vragen van leden. De edities worden samengesteld op basis van een groot aantal prioriteiten, welke door een redactievergadering worden gehanteerd. Deze vergadering bestaat uit de vaste medewerkers zoals in de colofon vermeld. Het aantal inzendingen is groter dan in een enkele editie van minimaal 48 pagina's is te verwerken. Hierdoor kan het voorkomen dat een inzending eerst na enige tijd kan worden gepubliceerd.

DE CLUB HEEFT BEHOEFTE AAN MEER LEDEN. WIJ WILLEN MEER AAN KUNNEN BIEDEN DAN NU AL HET GEVAL IS. WERF DAAROM EEN LID!

WILT U EEN PRIJSLIJST? STUUR EEN GEFRANKEERDE ENVELOP AAN HET REDACTIE-ADRES.

Een onafhankelijke jury kent jaarlijks een aantal aanmoedigingspremies toe aan auteurs van gepubliceerde artikelen in DE 6502 KENNER.



De 6502 KENNER is een uitgave van de KIM gebruikers Club Nederland.

Adres voor het inzenden van en reacties op artikelen voor DE 6502 KENNER:  
Willem L. van Pelt  
Jacob Jordaensstraat 15  
2923 CK Krimpen a/IJssel  
Tel.: 01807 - 19881

## Vaste medewerkers:

Willem L. van Pelt  
Gerard van Roekel  
Frans Smeehuijzen

## Freelance medewerkers:

Frans Bakx  
Rob Banen  
Fridus Jonkman  
Gert Klein  
Roger Langeveld  
Anton Muller  
Gert van Opbroek  
Ruud Uphoff  
Frans Verberkt  
Fred Behringer, Munchen

Gehele of gedeeltelijke overname van de inhoud van DE 6502 KENNER zonder toestemming van het bestuur is verboden. Toepassing van gepubliceerde programma's, hardware etc. is alleen toegestaan voor persoonlijk gebruik.

DE 6502 KENNER verschijnt 6 x per jaar en heeft een oplage van 500 exemplaren.

Copyright (C) 1986 KIM Gebruikers Club Nederland.

De voorpagina is de DOS65-controllerkaart, op CAD/CAM ontwikkeld door Adri Hankel/Erwin Viisschedyk.

In verband met auteurswetgeving en andere maatregelen op het gebied van bescherming van software kan de redactie geen aansprakelijkheid aanvaarden voor inzendingen. Inzendingen dienen afkomstig te zijn van de inzender, tenzij anders aangegeven.

INHOUDSOPGAVE DE 6502 KENNER NR. 42 FEBRUARI 1986

1. Uitnodiging Bijeenkomst Geldrop 4.
2. Van de redactie 3.
3. CBM64 TIPS EN TRUUKS 2.  
... Gerard van Roekel
4. General Introduction to the Pascal Language 5.  
Een algemene introductie omtrent de programmeertaal Pascal  
... Gert van Opbroek
5. Real-Time-Clock voor de JUNIOR 9.  
Software voor Clock-IC MC146818  
Zie publikatie in Elektuur nr. 258 van april 1985  
... M.J. Spithorst
6. OCTOPUS 65 15.  
Fout in publikatie FCU kaart Elektuur Special
7. Compressor met RUN voor de APPLE 20.  
Een onvolkomenheid in Ruud Uphoff's publikatie in  
DE 6502 KENNER nr. 25.  
... Frans Verberkt
8. Demonstratieprogramma voor het maken van grafieken 22.  
Basic-programma volgens op publikatie in editie 41, december 1985  
... Hans Bosch, Twente University of Technology
9. VIC 20 23.  
FORTH on the VIC  
... Fred Behringer, Munchen
10. OCTOPUS 65 24.  
Let it be  
Hardware-artikel met Basic-programma voor muzikale Elektuur-computer  
... Will Cuijpers
11. GRAF-V2.4 26.  
Routines voor de grafische kaart in DE 6502 KENNER nr. 36  
... J.J.A. Janssen
12. Programma "Competitiestanden" Handbalcompetitie 38.  
Basic-programma voor de Elektuur JUNIOR computer  
Deel 1: Inleiding/Documentatie competitiestanden  
... Gerard Keet
13. FORTH Screen Editor 45.  
Full-screen-editor  
... Fridus Jonkman
14. OCTOPUS 65 48.  
Word Processor Printer Version  
Changes and extensions of the Wordprocessor V2.2  
... Leif Rasmussen, Danmark
15. Basicode 49.  
Joystick software voor Basicode  
... W. van Dongen

ZEND UW PROGRAMMA'S OF HARDWARE BESCHRIJVINGEN NAAR DE REDAKTIE !!!  
ANDERE LEDEN MAAKT U ER BLIJ MEE, ANDERE LEDEN DOEN HET DAN OOK !!!

C B M 6 4 T I P S E N T R U U K S

HET SIMULEREN VAN EEN ESCAPE

Voor de meeste (niet commodore) printers wordt met de ESCAPE-TOETS de printer verteld dat er iets bijzonders moet gebeuren (zie de handleiding van diverse printers). Doch de commodore 64 bezit geen ESCAPE-TOETS. Toch is op twee manieren deze functie te realiseren. Ten eerste in een programma met de instructie CHR\$(27). Of in de direkt-mode door de SHIFT en : (dubbele punt) in te drukken. Beide manieren voeren een ESCAPE uit.

KASSETTE LAAD PROBLEMEN

Het kan weleens gebeuren dat u, nadat u wat gestoeid heeft met de computer, plotseling geen programma meer krijgt ingeladen. De kassette-rekorder blijft tot het einde van de band draaien. Soms is dit euvel te verhelpen door in de direkt-mode het volgende in te toetsen:

POKE0,47

Hiermede wordt bit 2 van adres 0 weer op 1 gezet. Adres 0 bit 2 is normaal altijd 1.

BEPAALED REGEL WISSEN

Door de volgende regel in te toetsen is het mogelijk om een regel te wissen

POKE 781,R : SYS 59903

De letter R is het regelnummer (0 t/m 24).

DIRECTORY TIPS

Wilt u het aantal vrije blokken zien zonder de gehele directory. Toets het volgende in:

LOAD "0:\$",8

Of alle titels welke met een K beginnen:

LOAD "0:\$K\*",8

Ook is het mogelijk om bijvoorbeeld alle titels van programma's te tonen die uit 4 tekens bestaan, waarvan de eerste letter een 'B' is en de vierde letter een 'N'. Toets dan het volgende in:

LOAD "0:\$B??N"

KASSETTE LOAD ERROR

Indien u met normale snelheid een programma SAVED wordt dit twee maal na elkaar op de band geplaatst. Na een load opdracht wordt de eerste kopie van het programma geladen en wordt daarna vergeleken met de tweede kopie op de band. Daarbij wordt byte voor byte met elkaar vergeleken. Als alles klopt krijgt u keurig READY terug. Indien er verschillen door de computer worden vastgesteld, dan krijgt u een LOAD ERROR te zien. U weet dan nog niet of de fouten in de eerste of de tweede kopie zaten. Geef LIST en kijk of de listing er goed uitziet. Indien ja, typ dan (zonder regelnummer):

POKE45, PEEK(831):POKE46, PEEK(832):CLR

Er is een redelijke kans dat uw programma dan in orde is. Save het programma opnieuw en vergeet niet daarna een 'VERIFY' te doen. Indien u na 'LIST' niets op het scherm ziet of een hele slechte listing, dan zat de fout waarschijnlijk in de eerste kopie. Probeer dan opnieuw te laden eventueel na de koppen schoongemaakt te hebben of een andere rekorder.



Van de redactie.

Lid zijn van onze club heeft een bijzondere betekenis. Lid zijn van onze club betekent niet alleen: op je hobbykamer spelen met je computer, het lezen van het clubblad, de service van de club genieten en de clubbijeenkomsten bijwonen om clubgenoten te ontmoeten. Het betekent vooral: meehelpen het doel van de club gestalte te geven, en dat is: 'kennissuitwisseling'. Dit begrip kennissuitwisseling omvat méér dan men zou verwachten. De vraag rijst immers: hoe doe je dat het meest efficiënt, hoe heb je er zelf het meest profijt van. De beantwoording van die vraag hangt nauw samen met de aard en de hoeveelheid van de middelen die nodig zijn om er gestalte aan te geven. Op het moment dat ik met een mederedakteur lid op mijn hobbykamertje zit en hem demonstreer hoe een bepaald software probleem is opgelost, dan is er sprake van kennissuitwisseling. Kom ik op een landelijke bijeenkomst en verneem ik van een ander lid hoe hij zijn floating point routines heeft geschreven, dan is er eveneens sprake van kennissuitwisseling. Er is ook sprake van als ik van een familielid in Canada per post een omvangrijke Basic listing voor zijn 8080-machine ontvang; er is altijd wel iets in te vinden dat je aandacht trekt en dat je een nieuw idee aanreikt. Of deze vormen van kennissuitwisseling efficiënt zijn, of je daarmee de goede dingen doet om de kennissuitwisseling goed te laten zijn, effectief bezig te zijn, dat betwijfel ik. We moeten namelijk niet vergeten dat we lid van een club zijn geworden om kennis te nemen, iets te willen leren, althans iets op willen steken van hetgeen anderen aan hardware of aan software hebben ontwikkeld en welke technieken men daarbij heeft gebruikt. Om die kennissuitwisseling te relateren aan dat wat we beogen, kunnen we niet volstaan met vrijblijvende ontmoetingen, al zal niemand ontkennen dat ook die een eigen waarde, een eigen gezelligheid hebben. We zullen echter andere middelen nodig hebben, waarvan geld er een is. Geld kan, indien in voldoende mate aanwezig, en indien goed beheerd, een stevige fundering leggen onder het meest efficiënte object van kennissuitwisseling sinds eeuwen, het geschreven woord. En bij ons is dat het clubblad, omdat dat ieder van de clubleden bereikt. Berichtgeving via de schijf is een middel, net als berichtgeving via modem. Maar, gezien de incompatibiliteit met andere systemen, niet algemeen toepasbaar. Het clubblad biedt iedereen eenieder kennis aan, en heeft een veel meer dokumenterend effect.

Een clubblad kost veel geld, meer dan driekwart van de contributie. Om het goed te laten functioneren is er veel geld nodig. Hoe kom je daaraan? Heel eenvoudig. Door veel leden. Hoe kom je daaraan? Heel eenvoudig. Door veel bekendheid te geven aan het bestaan van je club. Hoe kan de club dat zelf doen? Heel eenvoudig. Door veel reclame te maken? Hoe kun je veel reclame maken? Door veel geld te hebben. Hoe kom je dan aan veel geld? Heel eenvoudig. Door veel leden te werven. En zo kunnen we nog uren met elkaar bezig zijn, zonder dat er iets gebeurt. Vandaar dat de aandacht eens gevraagd wordt voor de verantwoordelijkheid die we als clublid te dragen hebben. We moeten onze schouders eronder zetten om het ledenaantal uit te breiden, domweg omdat er geld nodig is. Want, halen we met z'n allen nieuwe leden binnen, dan halen we ook meer kennis binnen. En vele nieuwe leden hebben allang kennis gemaakt met de manier waarop uw redactie tracht die kennis beschikbaar te krijgen voor u! We zijn er dan echter nog lang niet. U als clublid kan een concrete bijdrage leveren aan de vulning van het clubblad. Want een clubblad zonder publikaties, dat heeft naar alle waarschijnlijkheid niet het effect dat de leden 'hoera' zullen roepen.

Ben je als redakteur voldaan achterover in je stoel gaan leunen als je de komende editie weer voldoende copy hebt, dan zul je er niet in slagen het clubblad de betekenis mee te geven die het behoort te hebben: een waardevol en lezenswaardig geheel van boeiende informatie, die probeert een weergave te zijn van hetgeen in de club leeft. Het is echter niet, zoals al zoveel verkondigd, het juiste uitgangspunt om van de veronderstelling uit te gaan dat de redactie het blad maar vol moet schrijven. Het is dan immers niet een representatie van hetgeen in de club leeft, maar de uiterlijk verschijningsvorm van hetgeen de redactie of een of meer leden daaruit beweegt. Prima natuurlijk, als het niet anders zou kunnen. Dat het anders kan, dat is de afgelopen jaren bewezen, gezien de enthousiaste bijdragen welke van de leden werden ontvangen. Men is ervan overtuigd geraakt dat het inzenden van al datgene wat men aan hardware of software ontwikkelde een bijdrage leveren kan aan het verhogen van de kwaliteit van het clubblad, door de mogelijkheid die dan ontstaat om variatie aan te brengen. Het is die variatie die uitdagend kan werken op onze prikkelende behoefte van elkaar te leren, nieuwe ideeën op te doen, de aandacht te trekken tot in verre landen. Ik ken menig clubblad. Goede en minder goede. Toch geloof ik dat er te weinig mensen zijn die er bijdragen aan leveren. Het gevolg is vaak dat het op den duur niet meer boeit. Dat is anders als leden van de club tegelijkertijd medewerker van het clubblad zijn, ieder op zijn eigen wijze, maar ieder bewust bezig met het gegeven dat hij medeverantwoordelijkheid draagt voor het voortbestaan van het clubblad. Leveren clubleden niet meer in, dan houdt het clubblad op te bestaan. Het kan niet genoeg gezegd. Door het inzenden van al datgene wat je ontwikkelde ontstaat vaak ook een hechte persoonlijke relatie met de redactie. Het heeft het voordeel dat men nog eens op een idee komt, aan een idee wordt geholpen, en soms wordt je gevraagd iets te doen of te ontwikkelen.

Er is ook nog iets anders te wensen. Jaren geleden was er niet veel voor nodig om uit de hoek van de zelfbouwers van de KIM en de SYM-1 copy te krijgen voor het clubblad. Andere informatie of dokumentatiemateriaal voor een goed functioneren van de redactie, waar men toch zelf belang bij heeft, werd niets eens meer gevraagd, het werd spontaan toegezonden. Moest men hardware maken, tot een compleet gedocumenteerd disk operating system toe, geen nood! Software nodig? Twaalf kilobyte? Geen nood! Het heeft zijn tijd nodig, maar het komt.

Toch viel het niet mee om uit de hoek van de andere computer-bezitters publiceersbaar materiaal te verkrijgen. Een enkele keer lukte het, maar het bleef te weinig. Daarom blijf ik hameren op het thema: lever allemaal wat in. Het clubblad zal erdoor verlevendigen. Doe als die APPLE-bezitters die enige tijd geleden zijn ingegaan op mijn oproep om mee te doen, om zelf verantwoordelijkheid te dragen voor het clubblad, door steeds in te zenden. Inmiddels zijn de resultaten van die contacten in het clubblad zichtbaar geworden. In de copy-buffer zit echter nog steeds niet genoeg materiaal dat mij de zekerheid geeft dat we ook anderen hun kennis kunnen doorgeven. Kijk nog maar eens goed. U hebt vast wel een en ander liggen. Vraag je niet af of het wel goed genoeg is. Het is altijd goed genoeg om er een contact met de redactie mee op te bouwen. Het kan het begin zijn van een prettige samenwerking. Het kan het begin zijn van een bijdrage aan een nog beter clubblad, Uw eigen clubblad!



## UITNODIGING BIJEENKOMST

Datum : zaterdag 15 MAART 1986  
 Lokatie : Gemeenschapshuis "De Zes Gehuchten"  
 Papenvoort 10 te Geldrop (bij Eindhoven)

ENTREEPRIJS: FL. 10,==

### Reisroute:

- per auto

- vanaf snelweg E3 Eindhoven - Venlo  
 Afslag Geldrop bij Golden Tulip hotel.  
 Bij eerste verkeerslicht richting centrum linksaf slaan (Emopad). Bij kruising na ca 800 m. linksaf slaan (Hoog Geldrop). Deze weg blijven volgen tot gebouw naast kerk aan rechterzijde.

- vanuit rondweg Eindhoven  
 Afslag Geldrop bij DAF showroom (Eindhovense weg). Weg volgen tot tweede verkeerslicht voor spoorwegaadukt. Rechtsaf slaan en meest rechtse weg inslaan (Papenvoort). Deze weg blijven volgen tot gebouw voor kerk aan linkerzijde.

- vanuit Nuenen - Helmond  
 In Geldrop richting Eindhoven. Bij 1e verkeerslicht voorbij spoorviadukt linksaf (voorsorteren). Verder als bij route vanuit rondweg Eindhoven.

- per trein - Stoptrein Eindhoven Weert  
 In Geldrop direct na uitgang station links voetgangerstunnel door (Tournooiveld). Weg oversteken en rechtsaf gaan. Eerste laan links (Hertogenlaan). Na bocht eerste straat rechts (Gildestr.) Eerste straat links (Diepenvaart). Gebouw aan einde links (naast kerk van Zesgehuchten).

Lunchpakket zelf meenemen.  
 Consumpties tegen betaling.

### Programma:

- 09.30 Zaal open met koffie
- 10.15 Opening door de voorzitter
- 10.30 Lezing over DOS65
  - ... Full screen editor
  - ... I/O redirect mechanisme
 door Erwin Visschedijk en Adri Hankel
- Lunchpauze
- 13.00 MARKT
- 13.15 INFORMEEL GEDEELTE
 

In dit deel kunnen alle leden hun zelf ontwikkelde hardware en software demonstreren.

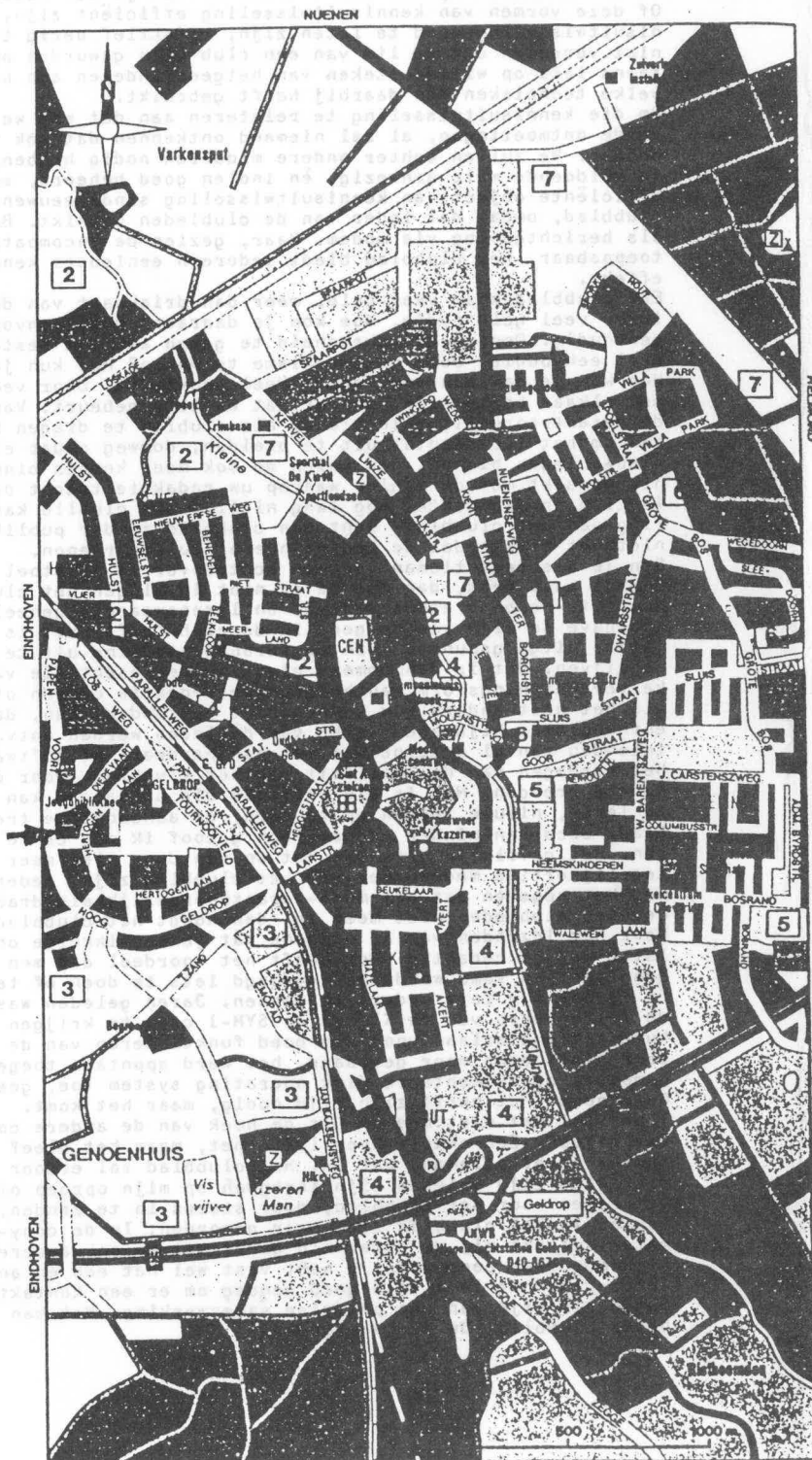
Het illegaal gebruik en het kopiëren van software waarop enige vorm van rechten rusten, ook die van de club, is ten strengste verboden!

BRENG UW EIGEN SYSTEEM MEE !!!

Het aanbieden van overbodige spullen is op eigen tafel(s) te regelen.
- 17.00 SLUITING.

De redactie roept eenieder op via de radio ontvangen Basicode programma's in te sturen.

Geef U bij de redactie op voor het maken van illustraties, vertalingen naar het engels, of illustratieve tekeningen.



HEEZE/A2



## GENERAL INTRODUCTION TO THE PASCAL LANGUAGE.

By: Gert van Opbroek  
Bateweg 68  
2481 AN Woubrugge.

Tel 01729-8636

### 1. Introduction.

In the near future we are going to publish some programs in the PASCAL language and therefore I was asked to write a general introduction to this language.

PASCAL was developed by Niklaus Wirth at the EHT in Zuerich, Switzerland in 1971, mainly for educational purposes. It is a so called ALGOL like language what means that the grammatics of the language resembles ALGOL strongly.

Although I have never worked with PASCAL on a 6502 microprocessor I have about 5 years experience in PASCAL on Cyber, PDP-11 and CP/M systems and I want to write down some of my experiences in this article to help other members of the KIM-CLUB to understand some of the features of PASCAL.

PASCAL is used for very different problems in the industry, research laboratories and telecommunication. Some of the problems for which PASCAL is used are:

- The control of measurement equipment at research laboratories.
- Data reduction and analysis at research laboratories. In the past FORTRAN was mostly used for this purpose but since students have to learn PASCAL during their education, PASCAL is more and more used for these applications.
- Production registration and stock control for a dutch milk factory.
- A complete system for system development, program specification, generation and documentation is programmed in PASCAL.
- The Dutch Post, Telephone and Telegraph Compagny (PTT) uses PASCAL for most of their new projects.

You see, that although PASCAL is developed for educational reasons, it has grown out to a professional language that is used in many fields of research and industry.

Many of the implementations of PASCAL conform to the ISO standard what means that these versions support all of the standard possibilities of PASCAL.

### 2. Compilers and interpreters.

Pascal uses a compiler in stead of an interpreter like most Basic and Comal versions. This means that when you wrote a program and want to run it, you have to translate (compile) this program with a so called compiler. This compiler translates the Pascal instructions to a program in a lower language, for instance in machine instructions or P-code instructions. When you run your program this lower language program is executed.

Interpreters, like most Basic versions, execute the high level instructions directly, that means, for every high level instruction there is a small (or large) subroutine that executes the high level instruction. That means that at runtime the complete source of the program is in main memory and when you want to change the program, all you have to do is type in the changes and rerun the program.

Because in this world, you get nothing for nothing, the execution speed of an interpreted language is much lower than that of a compiled language. The mean reason for this is the fact that the code, made by a compiler can be optimized.

It is not true that because Basic is interpreted, it has fewer possibilities than Pascal. BBC-Basic has much of the Pascal possibilities and the language Lisp, which is also interpreted has, in my opinion, even more possibilities than Pascal.

So the main reason to use a compiler for languages like Pascal, Fortran and so on is the execution speed and the compact code that is generated by the compiler.

### 3) Comparison of the possibilities of Pascal and Basic.

When we compare Pascal with Basic, there are many differences; most of them are in favour of Pascal. I shall give a brief overview of the most significant ones.

#### A) Variables.

Basic supports three types of basic, so called scalar, variable types:

INTEGER, REAL and STRING.

In Pascal there are the scalar types:

BOOLEAN, INTEGER, REAL and CHARACTER

- A boolean is a logical variable with the values TRUE and FALSE and is used in logical expressions.
- Character variables have values defined by the ASCII character set. Some Pascal versions have extra character values, for instance graphical characters.

You can also define your own scalar type by mentioning all the values of the type, for instance:

TYPE COLOUR = (RED, BLUE, YELLOW, WHITE, BLACK);

Basic has only one structured type: the one or more dimensional array of scalar types.

In Pascal we have ARRAYS, SETs and RECORDs.

The ARRAY in Pascal can have all scalar types, except real, as index type. The range of these indexes can be the full range of the index type. In more dimensional arrays one or more different index types with different ranges can be used.

An array can be PACKED or unpacked. A PACKED ARRAY means that for every item the least possible memory space is used. The only practical consequence is that a PACKED ARRAY of CHAR can be read from one wrote to a textfile with only one READ or WRITE statement. For all other types of ARRAYS read and write must be done element by element.



A two dimensional array is defined as an array of an array. So:

```
TYPE MATRIX = ARRAY[1..10,1..100] OF INTEGER
```

is a shorthand notation for

```
TYPE MATRIX = ARRAY[1..10] OF ARRAY[1..100] OF INTEGER.
```

In Pascal we can make arrays of any type; so we can make ARRAYS of CHARACTER, ARRAYS of ARRAYS, ARRAYS of COLOUR, ARRAYS of RECORDS .....

An ARRAY of CHARACTER is often used as a string. Unfortunately ISO Pascal offers no routines for string manipulation like VAL\$, MID\$, LEFT\$ and RIGHT\$ in Basic. I find this the only advantage of Basic above Pascal. We even can't copy a string with length L in a string with length M (<) L in one single assignment. Fortunately, many Pascal versions offer procedures for string manipulation.

A SET is a set (verzameling in dutch) of some scalar type (except real). For instance a SET of CHAR or a SET of INTEGER. A set can only have a limited number of members (often 255). A SET is often used for logical expressions for instance:

```
IF c IN ['A'..'Z','0'..'9'] THEN .....
```

For SETs Pascal has the operators: + (union, vereniging in dutch), - (set difference, this means all the members of A which are not members of B), \* (intersection, doorsnede in dutch) and IN (member of).

The most powerfull structured type in Pascal is the RECORD type. In a RECORD we can bundle different information in one variable. The best way to show this is with an example:

```
TYPE PERSON =
  RECORD OF
    FIRST_NAME : PACKED ARRAY[1..10] OF CHAR;
    FAMILY_NAME : PACKED ARRAY[1..30] OF CHAR;
    AGE : INTEGER;
    SEX : (MALE,FEMALE);
  END;
```

When we have an variable VAR\_P of type PERSON we can access the fields in this record by:

```
VAR_P.FIRST_NAME := 'PIET';
VAR_P.SEX := MALE;
```

We can also use the WITH statement:

```
WITH VAR_P DO
  BEGIN
    FAMILY_NAME := 'PIETERSEN';
    FIRST_NAME := 'JOOP';
  END;
```

A RECORD can have a so called variant part what means that the contents of the record can vary as function of one of the variables:

```
TYPE PERSON =
  RECORD OF
    FIRST_NAME : PACKED ARRAY[1..10] OF CHAR;
    FAMILY_NAME : PACKED ARRAY[1..30] OF CHAR;
    CASE SEX : (MALE,FEMALE) OF
      MALE : (BEARD : BOOLEAN;
              MARRIED : BOOLEAN);
      FEMALE : (CHILDREN : INTEGER);
    END;
```

The last special type we have in Pascal is the pointer (^) type. A pointer variable contains the address of the variable (or begin address in the case of structured types). This means that with pointer types we have a form of indirection. When we put in a RECORD one or more pointers to the same type of RECORD we can make linked lists, trees and networks. When there is enough interest in our club, I can write something about pointer types lists and trees in a next issue of the 6502-kenner.

You see that Pascal offers many more types of variables than Basic. I think that this is the most significant difference between Basic and Pascal.

## B) Expressions.

One of the most significant differences between Basic, Comal and Pascal is the fact that standard Pascal can not raise to a power like the ^ operator in Basic, Comal. When we need something like that, we have to write our own function for that or use the EXP function.

For square we can use the function SQR and for square-root we can use SQRT. As discribed above, Pascal has no string operations and supports SET operations.

## C) Assignment.

In Pascal assignment is done by: :=, just like in Comal. In Pascal, variables of the same type can be assigned to each other by one statement. So we can assign RECORDs and ARRAYS by one statement. Because a two dimensional array is defined as ARRAY of ARRAY, we can assign a complete column in a matrix by one single instruction:

```
TYPE MATRIX = ARRAY[1..10,1..30] OF INTEGER;
```

```
VAR M : MATRIX;
    R : ARRAY[1..30] OF INTEGER;
```

```
BEGIN
  .....
  M[5] := R;
  .....
END.
```

## D) Statements.

Most Basic versions have no compound statements. The only construction that is supported in Basic that looks like a compound statement is:

```
IF test THEN s1 : s2 : s3 ... ELSE s4 : s5 : s6
```

In Comal we have constructions like:

```
IF test THEN
  s1
  s2
  s3
ELSE
  s4
  s5
  s6
ENDIF
```

Pascal has a much more general solution for this problem:

Every block of statements, beginning with BEGIN and ending with END is treated as one statement. We can also have nested statements:

```
BEGIN
  s1
  s2
  BEGIN
    s3
    s4
  END;
  s5
END;
```

## E) Control structures.

Pascal supports five types of control structures:

- The IF-THEN-ELSE structure in the form  
IF boolean expression  
THEN  
  (compound) statement  
ELSE  
  (compound) statement  
in which the ELSE part is optional.

Remark: Before an ELSE you may not use a semicolon (;). This is one of the most frequently compiling errors in my programs. The reason for this is the fact that most programmers use the semicolon as a terminator in stead of a separator. ; after the IF-part means that there is no ELSE-part in the statement.

- The CASE statement.

The CASE statement is used in the construction:

```
CASE case selector OF
  v1 : (compound) statement ;
  v2 : (compound) statement ;
  v3 : (compound) statement ;
  .
  .
  .
END
```

The case selector must be a scalar type (except REAL). When you have one statement for two or more values of the case selector you can separate these values by a comma :

```
CASE case selector OF
  v1 : (compound) statement ;
  v2,v3 : (compound) statement ;
  v4 : (compound) statement ;
  v5 : ;
END
```

Behind v5 in the above example you see a so called empty statement. This is done for the reason that Pascal demands that for every value that the case selector can have during run-time there is a case label. Fortunately have most Pascal versions the default label OTHERWISE (without colon) what means for every other value of the case selector:

```
CASE case selector OF
  v1 : s1 ;
  v2 : s2 ;
  v3,v4 : s4 ;
  OTHERWISE s5 ;
END
```

- The WHILE DO statement. This statement has the form:

WHILE boolean expression DO (compound) statement.

The (compound) statement is executed until the boolean expression is false. This means that if the expression is already false in the first pass, the expression is not executed.

- The REPEAT UNTIL statement in the form:

```
REPEAT
  statement 1 ;
  statement 2 ;
  .
  .
  .
  statement n
UNTIL boolean expression
```

The statements between REPEAT and UNTIL are executed until the boolean expression is true. These statements are executed at least once.

- The FOR TO/DOWNTO DO statement. This is the indexed repetitive statement. It has two forms:

```
FOR index variable = start value TO end value DO
  (compound) statement
FOR index variable = start value DOWNTO end value DO
  (compound) statement
```

The first form has a positive increment, the second has a negative increment.

The index variable can be every scalar type (except REAL) and the increment is always one. This means that the construction with TO takes the successor of the index variable and the construction with DOWNTO takes the predecessor of the index variable. Pascal does not know FOR loops with something like the STEP statement in BASIC.

## F) PROCEDURES and FUNCTIONS.

In Basic we have the so called subroutines. That are parts of a program which are called with GOSUB and which end with RETURN. There is no parameter passing to subroutines. In COMAL there is a way to pass parameters to procedures. This is always a call by reference. In Pascal we have all we want to have in the procedures; parameter passing by value and by reference, local variables and procedure or function parameters. Recursion is one of the standard possibilities of Pascal what means that a procedure (or function) can call it self. In the next section I shall describe these features of Pascal more in detail.

In Basic one can define a function by:

```
DEF FNnaam(parameters) = expression
```

In Pascal we define a function just like a procedure.

All we have to do is:

- Change PROCEDURE in FUNCTION
- Declare the result type; this must be a scalar or a pointer type
- Do an assignment to the function.

## G) Standard PROCEDURES and FUNCTIONS.

Most of the standard procedures and functions in Basic have their counterpart in Pascal (except string functions).



Pascal has formatted output to textfiles although I find it a bit clumsy to use.

In Pascal we do not have a PEEK or POKE although we often can assign a variable name to an absolute address.

In Pascal we have a function NEW which is used with a pointer variable as parameter and gives dynamic a new (empty) record to which this parameters points.

For two standard functions I shall give a brief explanation:

- SUCC(x) : gives the successor of the scalar x within the ordered set of values of x. This means SUCC(10) = 11 ; SUCC('P') = 'Q' and from the example in section 3A SUCC(YELLOW) = WHITE.
- PRED(x) : gives the predecessor of x within the ordered set of x. So PRED(10) = 9 ; PRED('P') = 'O' and PRED(YELLOW) = BLUE.

#### 4. PROCEDURES and FUNCTIONS in Pascal.

As remarked in section 3, Pascal offers much more possibilities in procedure and function calls than a language like Basic or Comal.

The most significant differences are local variables, parameters and recursion.

- A) Local variables. Pascal makes a difference between local and global variables. Every variable in Pascal has to be declared. If it is declared in the main program, that means after the PROGRAM statement and before the first BEGIN of the main program, it is a so called global variable. This means that the variable can be used in the main program and every procedure or function that does not redeclare the variable. If a variable is declared in a procedure or function, then it is a so called local variable. This means that the variable can only be used within that procedure or function. If there is a global variable with the same name as the local variable then only the local variable can be used.

Note: Mostly we talk about global and local variables; in Pascal it is better to talk about global and local names because we can have also local types, procedures, labels and constants which differ from the global ones.

#### B) Parameters.

In general, we can have three types of parameter passing:

- call by value
- call by reference
- call by name

Pascal offers the first two, Basic only the call by value (in function calls) and COMALX.KGN only the second in procedure calls. I know only one language that supports the call by name: ALGOL.

In call by value, the value of the parameter is passed to the procedure or function. In Pascal it is programmed as follows:

```
PROCEDURE PROC(P : INTEGER) ;
VAR ..... { local variables }
```

```
BEGIN { of procedure PROC }
```

```
.
```

```
END ; { of procedure PROC }
```

In call of PROC we can substitute the name of an INTEGER or an integer value:

```
PROC(1) ; ..... PROC(2) ;
```

If we use the word VAR before the definition of the procedure in its definition, we use a call by reference. This means that the reference (address) of the variable is passed to the procedure. This means also that we always have to pass a variable (for instance I in the previous example).

In call by name, the name of the variable is passed to the procedure. This is not the same as call by reference although it mostly gives the same result. If the name of the parameter is given to an other variable in the procedure, the reference of the parameter is changed. A very good example of this feature is the so called Jensen's device in Algol-60:

```
PROCEDURE SUM(N,J,A,S); VALUE N; INTEGER N,J; REAL A,S;
BEGIN S := 0 ;
  FOR J := 1 STEP 1 UNTIL N DO S := S + A
END;
```

The call of SUM is : SUM(M,H,G[H],SS) ;

In this procedure the NAME A is first assigned to the variable G[1], then to G[2] and so on until G[M].

In Pascal a procedure can have zero, one or more parameters of different types. Each of them can be a value or a reference parameter.

In Pascal, we can also pass complete procedures or functions as parameter. I shall only give an example of this feature:

```
FUNCTION ZERO(X,Y : REAL;FUNCTION F : REAL) : REAL ;
{ gives the argument of F between X and Y for which
  F(A) = 0
  ..... }
```

We can call this function as follows:

```
WRITELN('PI/2 =',ZERO(COS,0,2)) ;
```

Note: The algorithm of ZERO can be found in:

Jensen,Wirth : Pascal User Manual and Report.

#### C) Recursion.

In Pascal a procedure or function can call another procedure or function or itself. The only condition is that the name of the called procedure or function is known at the place where it is called. In general this means that the definition of the called procedure or function has to be made above the calling statement in the program.

CLOCK

JUNIOR MICRO-ADE 6502 KENNER

20:29:58 PAGE 01

0010: REAL-TIME-CLOCK VOOR DE JUNIOR

0020:

0030:

0040:

0050:

0060:

0070:

0080:

0090:

0100:

0110:

0120:

0130:

0140:

0150:

0160:

0170:

0180:

0190:

0200:

0210:

0220:

0230:

0240:

0250:

0260:

0270:

0280:

0290:

0300:

0310:

0320:

0330:

0340:

0350:

0360:

0370:

0380:

0390:

0400:

0410:

0420:

0430:

0440:

0450:

0460:

0470:

0480:

0490:

0500:

0510:

REAL-TIME-CLOCK VOOR DE JUNIOR

Onderstaande routines zijn geschreven voor de Real-Time-Clock, zoals beschreven in Elektuur nr. 258 (april 1985). Aangezien de schakeling in standby-bedrijf met de laagste kristalfrekwentie ook het laagste stroomverbruik heeft, heb ik een 32.768 KHz kristal toegepast. Als men dit kristal (zit ook in horloges) toepast moet C3 vergroot worden tot 470 pF a 1 nF, anders oscilleert het zaakje niet. Het in de schakeling toegepaste clock-IC (MC146818) heb ik ingesteld op BCD uitlezen, omdat MICRO-ADE dan ook vrij simpel gewijzigd kan worden om te werken met de Real-Time-Clock. Bij een koude start van M.A. hoeft dan niet meer de datum ingetikt te worden en de PASS 2 en TABLE-listings worden dan voorzien van de datum en de tijd in de kop van de listing.

M. J. Spithost

Weth. Huismanlaan 51

9902 LP Appingedam

B400

CLOCK ORG \$B400

PAGE ZERO ADRESSEN

E4 00	ADRES *	\$00E4	Buffer voor klok-adres
E5 00	DATA *	\$00E5	Buffer voor klok-data
E6 00	MEPNT *	\$00E6	Pointer voor PRINT-routine
E8 00	SAVNIB *	\$00E8	Buffer voor GETVAL-routine

MONITOR SUBROUTINES

E8 11	CRLF *	\$11E8	Nieuwe regel
F3 11	PRSP *	\$11F3	Druk spatie af
8F 12	PRBYT *	\$128F	Print byte als 2 ASCII-kar.
AE 12	RECCHA *	\$12AE	Haal toetswaarde
34 13	PRCHA *	\$1334	Druk karakter af

MC146818 ADRESSEN

70 18	CLOCKA *	\$1870	Klok adres
71 18	CLOCKD *	\$1871	Klok data

WRITE schrijft op de juiste wijze data naar de real-time-clock

WRITE	LDAZ	DATA	Data voor klok in accu
	LDXZ	ADRES	Klok adres in X-reg.
	STX	CLOCKA	klok adres naar klok
	STA	CLOCKD	Data naar klok
	RTS		



```

0520:                                READ leest op de juiste wijze data uit
0530:                                de real-time-clock
0540:
0550: B40B A6 E4      READ  LDXZ  ADRES  Klok-adres in X-reg.
0560: B40D 8E 70 18    STX    CLOCKA Adres naar klok
0570: B410 AD 71 18    LDA    CLOCKD Data in accu
0580: B413 85 E5      STAZ  DATA  Data in buffer
0590: B415 60        RTS
0600:
0610:                                POLL wacht tot er kan worden gelezen uit
0620:                                1 van de 10 tijd en datum registers
0630:
0640: B416 A2 0A      POLL  LDXIM $0A  Reg. $0A bevat UIP-bit
0650: B418 8E 70 18    NOTSET STX  CLOCKA
0660: B41B AD 71 18    LDA    CLOCKD Haal inhoud van reg. $0A
0670: B41E 10 F8      BPL    NOTSET Wacht tot bit 7 geset wordt
0680: B420 8E 70 18    SET    STX  CLOCKA
0690: B423 AD 71 18    LDA    CLOCKD Haal inhoud van reg. $0A
0700: B426 30 F8      BMI    SET    Wacht tot bit 7 gereset wordt
0710: B428 60        RTS
0720:
0730:                                PRINT drukt een string af
0740:
0750: B429 68          PRINT  PLA      Haal terugkeeradres van
0760: B42A 85 E6      STA      MEPNT  stapel en berg het op
0770: B42C 68          PLA
0780: B42D 85 E7      STA      MEPNT  +01
0790: B42F E6 E6      PRTA  INC      MEPNT
0800: B431 D0 02      BNE      PRTB
0810: B433 E6 E7      INC      MEPNT  +01
0820: B435 A0 00      PRTB  LDYIM $00  Haal karakter en
0830: B437 B1 E6      LDAIY MEPNT  druk het af
0840: B439 C9 03      CMPIM $03  EOT-teken ?
0850: B43B F0 06      BEQ      PRTC
0860: B43D 20 34 13    JSR      PRCHA  /
0870: B440 4C 2F B4    JMP      PRTA
0880: B443 A5 E7      PRTC  LDA      MEPNT  +01 Herstel terugkeeradres
0890: B445 48          PHA
0900: B446 A5 E6      LDA      MEPNT
0910: B448 48          PHA
0920: B449 60        RTS
0930:
0940:                                Met de subroutine OKNOK kan de inhoud
0950:                                van een register gewijzigd worden
0960:
0970: B44A 20 29 B4    OKNOK JSR      PRINT
0980: B44D 43          =      'C
0990: B44E 4F          =      'O
1000: B44F 52          =      'R
1010: B450 52          =      'R
1020: B451 45          =      'E
1030: B452 43          =      'C
1040: B453 54          =      'T
1050: B454 20          =      $20
1060: B455 3F          =      '?'
1070: B456 20          =      $20
1080: B457 28          =      '{

```

```

1090: B458 4A      = 'J DATA J'
1100: B459 2F      = ' /'
1110: B45A 4E      = 'N'
1120: B45B 29      = ' ('
1130: B45C 0D      = $0D
1140: B45D 0A      = $0A
1150: B45E 03      = $03
1160: B45F 20 AE 12 JSR RECCHA Haal toets
1170: B462 C9 4A    CMPIM 'J Waarde juist ?
1180: B464 F0 35    BEQ OKEND Zo ja, einde
1190: B466 20 29 B4 JSR PRINT
1200: B469 0D      = $0D
1210: B46A 0A      = $0A
1220: B46B 54      = 'T
1230: B46C 4F      = 'O
1240: B46D 45      = 'E
1250: B46E 54      = 'T
1260: B46F 53      = 'S
1270: B470 20      = $20
1280: B471 4A      = 'J
1290: B472 55      = 'U
1300: B473 49      = 'I
1310: B474 53      = 'S
1320: B475 54      = 'T
1330: B476 45      = 'E
1340: B477 20      = $20
1350: B478 57      = 'W
1360: B479 41      = 'A
1370: B47A 41      = 'A
1380: B47B 52      = 'R
1390: B47C 44      = 'D
1400: B47D 45      = 'E
1410: B47E 20      = $20
1420: B47F 49      = 'I
1430: B480 4E      = 'N
1440: B481 0D      = $0D
1450: B482 0A      = $0A
1460: B483 03      = $03
1470: B484 20 AE 12 GETVAL JSR RECCHA Haal toets
1480: B487 E9 2F    SBCIM $2F ASCII naar HEX
1490: B489 0A      ASLA      Schuif nibble naar
1500: B48A 0A      ASLA      hoogstwaardige bits
1510: B48B 0A      ASLA      =
1520: B48C 0A      ASLA      =
1530: B48D 85 E8    STAZ SAVNIB Even opbergen
1540: B48F 20 AE 12 JSR RECCHA Haal toets
1550: B492 E9 2F    SBCIM $2F ASCII naar HEX
1560: B494 05 E8    ORAZ SAVNIB Voeg nibbles samen
1570: B496 85 E5    STAZ DATA
1580: B498 20 00 B4 JSR WRITE Schrijf nieuwe waarde
1590: B49B 60      OKEND RTS      in register
1600:
1610: De subroutine ADJREG drukt de inhoud van
1620: een register af, vraagt of de inhoud juist
1630: is en wijzigt de inhoud eventueel
1640:

```



```

1650: B49C A5 E5      ADJREG LDAZ DATA Haal data van register uit buffer
1660: B49E 20 8F 12      JSR PRBYT en print het
1670: B4A1 20 E8 11      JSR CRLF Nieuwe regel
1680: B4A4 20 4A B4      JSR OKNOK Inhoud wijzigen ?
1690: B4A7 60          RTS
1700:
1710: De subroutine RESET herstelt het SET-bit
1720: waarna de klok weer gaat lopen
1730:
1740: B4A8 A2 0B      RESET LDXIM $0B
1750: B4AA 86 E4      STXZ ADRES
1760: B4AC 20 0B B4      JSR READ
1770: B4AF A5 E5      LDAZ DATA Lees registerinhoud
1780: B4B1 29 7F      ANDIM $7F Reset het SET-bit
1790: B4B3 85 E5      STAZ DATA
1800: B4B5 20 00 B4      JSR WRITE
1810: B4B8 60          RTS
1820:
1830: CLKINI initialiseert de real-time-clock
1840:
1850: B4B9 A9 8A      CLKINI LDAIM $8A Stop update-cyclus
1860: B4BB A2 0B      LDXIM $0B en set modes
1870: B4BD 86 E4      STXZ ADRES (o.a. BCD en 24-uurs mode)
1880: B4BF 85 E5      STAZ DATA
1890: B4C1 20 00 B4      JSR WRITE
1900: B4C4 A9 20      LDAIM $20 Tijdbasis = 32.768 KHz
1910: B4C6 A2 0A      LDXIM $0A
1920: B4C8 86 E4      STXZ ADRES
1930: B4CA 85 E5      STAZ DATA
1940: B4CC 20 00 B4      JSR WRITE
1950:
1960: B4CF A2 09      ADJUST LDXIM $09 Jaar-register
1970: B4D1 86 E4      STXZ ADRES
1980: B4D3 20 0B B4      JSR READ Lees jaarregister
1990: B4D6 20 29 B4      JSR PRINT
2000: B4D9 0D          = $0D
2010: B4DA 0A          = $0A
2020: B4DB 4A          = 'J'
2030: B4DC 41          = 'A'
2040: B4DD 41          = 'A'
2050: B4DE 52          = 'R'
2060: B4DF 20          = $20
2070: B4E0 3D          = '='
2080: B4E1 20          = $20
2090: B4E2 27          = ''
2100: B4E3 03          = $03
2110: B4E4 20 9C B4      JSR ADJREG Print reg.-inhoud en wijzig eventueel
2120: B4E7 A2 0B      LDXIM $0B Maandregister
2130: B4E9 86 E4      STXZ ADRES
2140: B4EB 20 0B B4      JSR READ Lees maandregister
2150: B4EE 20 29 B4      JSR PRINT
2160: B4F1 0D          = $0D
2170: B4F2 0A          = $0A
2180: B4F3 4D          = 'M'
2190: B4F4 41          = 'A'
2200: B4F5 41          = 'A'
2210: B4F6 4E          = 'N'

```

2220:	B4F7	44	=	'D	
2230:	B4F8	20	=	\$20	
2240:	B4F9	3D	=	'=	
2250:	B4FA	20	=	\$20	
2260:	B4FB	03	=	\$03	
2270:	B4FC	20 9C B4	JSR	ADJREG	Print reg.-inhoud en wijzig eventueel
2280:	B4FF	A2 07	LDXIM	\$07	Datumregister
2290:	B501	86 E4	STXZ	ADRES	
2300:	B503	20 0B B4	JSR	READ	Lees datumregister
2310:	B506	20 29 B4	JSR	PRINT	
2320:	B509	0D	=	\$0D	
2330:	B50A	0A	=	\$0A	
2340:	B50B	44	=	'D	
2350:	B50C	41	=	'A	
2360:	B50D	54	=	'T	
2370:	B50E	55	=	'U	
2380:	B50F	4D	=	'M	
2390:	B510	20	=	\$20	
2400:	B511	3D	=	'=	
2410:	B512	20	=	\$20	
2420:	B513	03	=	\$03	
2430:	B514	20 9C B4	JSR	ADJREG	Print reg.-inhoud en wijzig eventueel
2440:	B517	A2 06	LDXIM	\$06	Dagregister
2450:	B519	86 E4	STXZ	ADRES	
2460:	B51B	20 0B B4	JSR	READ	Lees dagregister
2470:	B51E	20 29 B4	JSR	PRINT	
2480:	B521	0D	=	\$0D	
2490:	B522	0A	=	\$0A	zondag = \$01
2500:	B523	44	=	'D	
2510:	B524	41	=	'A	
2520:	B525	47	=	'G	
2530:	B526	20	=	\$20	
2540:	B527	3D	=	'=	
2550:	B528	20	=	\$20	
2560:	B529	03	=	\$03	
2570:	B52A	20 9C B4	JSR	ADJREG	Print reg.-inhoud en wijzig eventueel
2580:	B52D	A2 04	LDXIM	\$04	Urenregister
2590:	B52F	86 E4	STXZ	ADRES	
2600:	B531	20 0B B4	JSR	READ	Lees urenregister
2610:	B534	20 29 B4	JSR	PRINT	
2620:	B537	0D	=	\$0D	
2630:	B538	0A	=	\$0A	
2640:	B539	55	=	'U	
2650:	B53A	52	=	'R	
2660:	B53B	45	=	'E	
2670:	B53C	4E	=	'N	
2680:	B53D	20	=	\$20	
2690:	B53E	3D	=	'=	
2700:	B53F	20	=	\$20	
2710:	B540	03	=	\$03	
2720:	B541	20 9C B4	JSR	ADJREG	Print reg.-inhoud en wijzig eventueel
2730:	B544	A2 02	LDXIM	\$02	Minutenregister
2740:	B546	86 E4	STXZ	ADRES	
2750:	B548	20 0B B4	JSR	READ	Lees minutenregister
2760:	B54B	20 29 B4	JSR	PRINT	
2770:	B54E	0D	=	\$0D	
2780:	B54F	0A	=	\$0A	



```

2790: B550 4D      =      'M
2800: B551 49      =      'I
2810: B552 4E      =      'N
2820: B553 55      =      'U
2830: B554 54      =      'T
2840: B555 45      =      'E
2850: B556 4E      =      'N
2860: B557 20      =      $20
2870: B558 3D      =      '='
2880: B559 20      =      $20
2890: B55A 03      =      $03
2900: B55B 20 9C B4 JSR  ADJREG Print reg.-inhoud en wijzig eventueel
2910: B55E A2 00     LDXIM $00 Secondenregister
2920: B560 86 E4     STXZ  ADRES
2930: B562 20 0B B4 JSR  READ Lees secondenregister
2940: B565 20 29 B4 JSR  PRINT
2950: B568 0D      =      $0D
2960: B569 0A      =      $0A
2970: B56A 53      =      'S
2980: B56B 45      =      'E
2990: B56C 43      =      'C
3000: B56D 4F      =      'O
3010: B56E 4E      =      'N
3020: B56F 44      =      'D
3030: B570 45      =      'E
3040: B571 4E      =      'N
3050: B572 20      =      $20
3060: B573 3D      =      '='
3070: B574 20      =      $20
3080: B575 03      =      $03
3090: B576 20 9C B4 JSR  ADJREG Print reg.-inhoud en wijzig eventueel
3100: B579 20 AB B4 JSR  RESET
3110:
3120: B57C 20 E8 11 DATIME JSR  CRLF Nieuwe regel
3130: B57F A2 06     LDXIM $06 Dagenregister
3140: B581 86 E4     STXZ  ADRES
3150: B583 20 16 B4 JSR  POLL Kan er gelezen worden?
3160: B586 20 0B B4 JSR  READ Lees dagregister
3170: B589 C9 01     CMPIM $01 zondag?
3180: B58B D0 0D     BNE  MONDAY
3190: B58D 20 29 B4 JSR  PRINT
3200: B590 5A      =      'Z
3210: B591 4F      =      'O
3220: B592 4E      =      'N
3230: B593 44      =      'D
3240: B594 41      =      'A
3250: B595 47      =      'G
3260: B596 03      =      $03
3270: B597 4C 03 B6 JMP  DATE
3280: B59A C9 02     MONDAY CMPIM $02
3290: B59C D0 0E     BNE  TSDAY
3300: B59E 20 29 B4 JSR  PRINT
3310: B5A1 4D      =      'M
3320: B5A2 41      =      'A
3330: B5A3 41      =      'A
3340: B5A4 4E      =      'N
3350: B5A5 44      =      'D

```

```

3360: B5A6 41      =      'A
3370: B5A7 47      =      'G
3380: B5A8 03      =      $03
3390: B5A9 4C 03 B6      JMP      DATE
3400: B5AC C9 03      TSDAY  CMPIM $03
3410: B5AE D0 0E      BNE      WEDNDY
3420: B5B0 20 29 B4      JSR      PRINT
3430: B5B3 44      =      'D
3440: B5B4 49      =      'I
3450: B5B5 4E      =      'N
3460: B5B6 53      =      'S
3470: B5B7 44      =      'D
3480: B5B8 41      =      'A
3490: B5B9 47      =      'G
3500: B5BA 03      =      $03
3510: B5BB 4C 03 B6      JMP      DATE
3520: B5BE C9 04      WEDNDY  CMPIM $04
3530: B5C0 D0 0F      BNE      THUDAY
3540: B5C2 20 29 B4      JSR      PRINT
3550: B5C5 57      =      'W
3560: B5C6 4F      =      'O
3570: B5C7 45      =      'E
3580: B5C8 4E      =      'N
3590: B5C9 53      =      'S
3600: B5CA 44      =      'D
3610: B5CB 41      =      'A
3620: B5CC 47      =      'G
3630: B5CD 03      =      $03
3640: B5CE 4C 03 B6      JMP      DATE
3650: B5D1 C9 05      THUDAY  CMPIM $05
3660: B5D3 D0 10      BNE      FRIDAY
3670: B5D5 20 29 B4      JSR      PRINT
3680: B5D8 44      =      'D
3690: B5D9 4F      =      'O
3700: B5DA 4E      =      'N
3710: B5DB 44      =      'D
3720: B5DC 45      =      'E
3730: B5DD 52      =      'R
3740: B5DE 44      =      'D
3750: B5DF 41      =      'A
3760: B5E0 47      =      'G
3770: B5E1 03      =      $03
3780: B5E2 4C 03 B6      JMP      DATE
3790: B5E5 C9 06      FRIDAY  CMPIM $06
3800: B5E7 D0 0E      BNE      SATDAY
3810: B5E9 20 29 B4      JSR      PRINT
3820: B5EC 56      =      'V
3830: B5ED 52      =      'R
3840: B5EE 49      =      'I
3850: B5EF 4A      =      'J
3860: B5F0 44      =      'D
3870: B5F1 41      =      'A
3880: B5F2 47      =      'G
3890: B5F3 03      =      $03
3900: B5F4 4C 03 B6      JMP      DATE
3910: B5F7 20 29 B4      SATDAY  JSR      PRINT
3920: B5FA 5A      =      'Z

```

## OCTOPUS 65

In de 1e uitgave van Elektuur Computing extra over de zelfbouwcomputer -OCTOPUS 65- is in de bouwbeschrijving van de FCU-kaart (Floppy Control Unit) een fout gemaakt waardoor de disk-drive niet aan de praat te krijgen is.

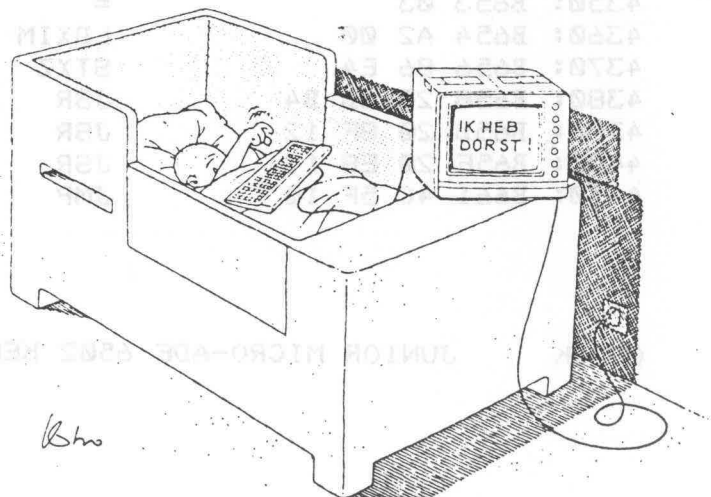
Bij de wijziging van de adressering van de kaart van C0xx naar E0xx is op pagina 35 met vette letters vermeld dat pen 10 moet worden doorverbonden.

In plaats hiervan moet men lezen:

-Niet pen 9 van IC7 (74LS138), maar pen 7 van IC7 moet met pen 8 van IC1 (74LS02) worden verbonden.

In het schema op pagina 33 is de wijziging wel goed getekend.

P.A. Zuiderwijk  
Jan Barendselaan 173  
2685 BS Poeldijk



3930:	B5FB	41	=	'A		
3940:	B5FC	54	=	'T		
3950:	B5FD	45	=	'E		
3960:	B5FE	52	=	'R		
3970:	B5FF	44	=	'D		
3980:	B600	41	=	'A		
3990:	B601	47	=	'G		
4000:	B602	03	=	\$03		
4010:	B603	20 F3 11	DATE JSR	PRSP		
4020:	B606	A2 07	DATEWS LDXIM	\$07	Datumregister	
4030:	B608	86 E4	STXZ	ADRES		
4040:	B60A	20 0B B4	JSR	READ		
4050:	B60D	20 8F 12	JSR	PRBYT		
4060:	B610	20 F3 11	JSR	PRSP		
4070:	B613	A2 08	LDXIM	\$08	Maandenregister	
4080:	B615	86 E4	STXZ	ADRES		
4090:	B617	20 0B B4	JSR	READ		
4100:	B61A	20 8F 12	JSR	PRBYT		
4110:	B61D	20 F3 11	JSR	PRSP		
4120:	B620	20 29 B4	JSR	PRINT		
4130:	B623	31	=	'1		
4140:	B624	39	=	'9		
4150:	B625	03	=	\$03		
4160:	B626	A2 09	LDXIM	\$09	Jarenregister	
4170:	B628	86 E4	STXZ	ADRES		
4180:	B62A	20 0B B4	JSR	READ		
4190:	B62D	20 8F 12	JSR	PRBYT		
4200:	B630	20 F3 11	JSR	PRSP		
4210:	B633	20 F3 11	JSR	PRSP		
4220:	B636	A2 04	LDXIM	\$04	Urenregister	
4230:	B638	86 E4	STXZ	ADRES		
4240:	B63A	20 0B B4	JSR	READ		
4250:	B63D	20 8F 12	JSR	PRBYT		
4260:	B640	20 29 B4	JSR	PRINT		
4270:	B643	3A	=	':		
4280:	B644	03	=	\$03		
4290:	B645	A2 02	LDXIM	\$02	Minutenregister	
4300:	B647	86 E4	STXZ	ADRES		
4310:	B649	20 0B B4	JSR	READ		
4320:	B64C	20 8F 12	JSR	PRBYT		
4330:	B64F	20 29 B4	JSR	PRINT		
4340:	B652	3A	=	':		
4350:	B653	03	=	\$03		
4360:	B654	A2 00	LDXIM	\$00	Secondenregister	
4370:	B656	86 E4	STXZ	ADRES		
4380:	B658	20 0B B4	JSR	READ		
4390:	B65B	20 8F 12	JSR	PRBYT		
4400:	B65E	20 E8 11	JSR	CRLF		
4410:	B661	4C 5F 10	JMP	\$105F		

CLOCK

JUNIOR MICRO-ADE 6502 KENNER

190985

20:43:55

PAGE 08



CLOCK JUNIOR MICRO-ADE 6502 KENNER 190985 20:54:39 PAGE 01

0010: 4160

CLOCK ORG \$4160

0020:

0030:

0040:

0050:

0060:

0070:

0080:

0090:

0100:

0110:

0120:

0130:

0140:

0150:

0160:

0170:

0180:

0190:

0200:

0210:

0220:

0230:

0240:

0250:

0260:

0270:

0280:

0290:

0300:

0310:

0320:

0330:

0340:

0350:

0360:

0370:

0380:

0390:

0400: 4160 20 F8 3E

0410: 4163 A2 07

0420: 4165 20 6D 25

0430: 4168 85 75

0440: 416A E8

0450: 416B 20 6D 25

0460: 416E 85 76

0470: 4170 E8

0480: 4171 20 6D 25

0490: 4174 85 77

0500: 4176 A2 04

0510: 4178 20 6D 25

0520: 417B 85 78

Datum en tijd-routines voor JUNIOR met MICRO-ADE

Onderstaande routines zorgen er voor dat de juiste datum en tijd afgedrukt worden tijdens een PASS 2 en een TABLE listing.

Hiervoor is de real-time-uP-klok uit Elektuur (april 1985) nodig die van tevoren op de voor deze routines juiste manier geprogrammeerd is. Register \$0B van de MC146818 moet ingesteld worden op BCD-uittezen.

M. J. Spithost,  
Weth. Huismanlaan 51,  
9902 LP Appingedam.

PAGE ZERO ADRESSEN

75 00 DATE \* \$0075 Datum en tijd buffer

Deze 6 adressen werden gebruikt om DDMMYY in op te slaan bij een koude start van MA. Omdat de datum en de tijd in BCD-code uitgelezen wordt, kunnen de adressen nu ook gebruikt worden om de tijd in op te slaan. De buffers zijn nodig omdat de data die uit de datum en tijdregisters gelezen wordt ongeveer 244 uS stabiel blijft.

Als er uitgelezen en direkt afgedrukt wordt, kunnen er foute gegevens op scherm en printer verschijnen.

MICRO-ADE SUBROUTINES

EE 2D	OUTSP *	\$2DEE	Druk spatie af
CD 2D	HEXOUT *	\$2DCD	Print byte als 2 ASCII-kar.
F0 2D	OUTCH *	\$2DF0	Druk karakter af
E7 2D	CRLF *	\$2DE7	Nieuwe regel

MC146818 ADRESSEN

70 18	CLOCKA *	\$1870	Klok adres
71 18	CLOCKD *	\$1871	Klok data

REDATI JSR POLL Wacht totdat er gelezen kan worden

LDXIM \$07 Datumregister

JSR READCK Uitlezen

STAZ DATE en in de buffer

INX Maandregister

JSR READCK

STAZ DATE +01

INX Jaarregister

JSR READCK

STAZ DATE +02

LDXIM \$04 Urenregister

JSR READCK

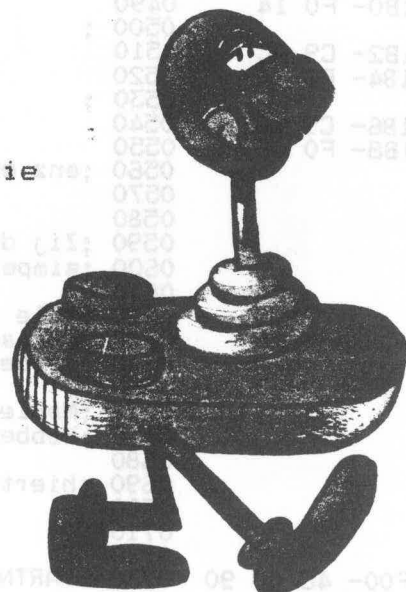
STAZ DATE +03

```

0530: 417D A2 02 LDXIM #02 Minutenregister
0540: 417F 20 6D 25 JSR READCK
0550: 4182 85 79 STAZ DATE +04
0560: 4184 A2 00 LDXIM #00 Secondenregister
0570: 4186 20 6D 25 JSR READCK
0580: 4189 85 7A STAZ DATE +05
0590: 418B B5 75 PRDATI LDAZX DATE Haal BCD-code uit buffer
0600: 418D 20 CD 2D JSR HEXOUT Druk af als 2 ASCII-karakters
0610: 4190 E8 INX X-reg. was voor 1e loop $00
0620: 4191 E0 03 CPXIM #03 Jaren al gehad ?
0630: 4193 D0 F6 BNE PRDATI Zo niet, doorgaan
0640: 4195 20 EE 2D JSR OUTSP Beetje ruimte
0650: 4198 20 EE 2D JSR OUTSP
0660: 419B B5 75 TIME LDAZX DATE Haal uren en minuten
0670: 419D 20 CD 2D JSR HEXOUT
0680: 41A0 A9 3A LDAIM ': Een paar puntjes er tussen
0690: 41A2 20 F0 2D JSR OUTCH
0700: 41A5 E8 INX
0710: 41A6 E0 05 CPXIM #05 Minuten al gehad ?
0720: 41A8 D0 F1 BNE TIME Zo niet, halen
0730: 41AA B5 75 LDAZX DATE Seconden
0740: 41AC 20 CD 2D JSR HEXOUT En druk af
0750: 41AF 60 RTS
0760:
0770: 3EF8 ORG $3EF8
0780:
0790: POLL wacht tot er kan worden gelezen uit
0800: 1 van de 10 tijd en datum registers
0810:
0820: 3EF8 A2 0A POLL LDXIM #0A Reg. #0A bevat UIP-bit
0830: 3EFA 8E 70 18 NOTSET STX CLOCKA
0840: 3EFD AD 71 18 LDA CLOCKD Haal inhoud van reg. #0A
0850: 3F00 10 F8 BPL NOTSET Wacht tot bit 7 geset wordt
0860: 3F02 8E 70 18 SET STX CLOCKA
0870: 3F05 AD 71 18 LDA CLOCKD Haal inhoud van reg. #0A
0880: 3F08 30 F8 BMI SET Wacht tot bit 7 gereset wordt
0890: 3F0A 60 RTS
0900:
0910: 254C ORG $254C Patch op GETDAY
0920:
0930: 254C B9 5E 25 GETDY1 LDAAY TEKST Haal karakter
0940: 254F 20 F0 2D JSR OUTCH Druk af
0950: 2552 C8 INY
0960: 2553 C0 0E CPYIM #0E Alles gehad
0970: 2555 D0 F5 BNE GETDY1 Zo niet, doorgaan
0980: 2557 20 60 41 JSR REDATI Lees datum en tijd en druk af
0990: 255A 20 E7 2D JSR CRLF Nieuwe regel
1000: 255D 60 RTS
1010: 255E 44 TEKST = 'D
1020: 255F 41 = 'A
1030: 2560 54 = 'T
1040: 2561 45 = 'E
1050: 2562 20 = $20
1060: 2563 41 = 'A
1070: 2564 4E = 'N
1080: 2565 44 = 'D

```

1090:	2566 20	=	\$20	
1100:	2567 54	=	'T	
1110:	2568 49	=	'I	
1120:	2569 4D	=	'M	
1130:	256A 45	=	'E	
1140:	256B 20	=	\$20	
1150:	256C 20	=	\$20	
1160:				
1170:				
1180:				
1190:				
1200:	256D 8E 70 18	READCK STX	CLOCKA	Adres naar klok
1210:	2570 AD 71 18	LDA	CLOCKD	Data in accu
1220:	2573 60	RTS		
1230:	2574 FF	=	\$FF	
1240:	2575 FF	=	\$FF	
1250:	2576 FF	=	\$FF	
1260:	2577 FF	=	\$FF	
1270:	2578 FF	=	\$FF	
1280:	2579 FF	=	\$FF	
1290:	257A FF	=	\$FF	
1300:				
1310:	3519	ORG	\$3519	Patch op TOPPRT
1320:				
1330:	3519 20 60 41	JSR	REDATI	Druk datum en tijd af
1340:	351C EA	NOP		
1350:	351D EA	NOP		
1360:	351E EA	NOP		
1370:	351F EA	NOP		
1380:	3520 EA	NOP		
1390:	3521 EA	NOP		
1400:	3522 EA	NOP		
1410:	3523 EA	NOP		
1420:	3524 EA	NOP		
1430:				
1440:	3714	ORG	\$3714	Patch op PRTS2
1450:				
1460:	3714 20 60 41	JSR	REDATI	Druk datum en tijd af
1470:	3717 EA	NOP		
1480:	3718 EA	NOP		
1490:	3719 EA	NOP		
1500:	371A EA	NOP		
1510:	371B EA	NOP		
1520:	371C EA	NOP		
1530:	371D 20 EE 2D	JSR	OUTSP	Nog een spatie





ISM L

```

0010 ;COMPRESSOR MET RUN
0020
0030 ; Frans Verberkt
0040 ; Hillekensacker 12-10
0050 ; 6546 KG NIJMEGEN
0060
0070 ; Tel 080 - 779555
0080
0090
0100 ;In de BASIC PROGRAM COMPRESSOR geschreven door Ruud Uphoff
0110 ;in 6502 KENNER nr.25 blz.10 t/m 21 is een onvolkomenheid ontdekt:
0120
0130 ;APPLE kent ook het statement RUN, welke gebruikt kan worden met verwijzing
0140 ;naar een regelnummer zoals b.v. RUN100 of RUN 550
0150
0160 ;Laat men de compressor hierop los, dan gaat er in de meeste gevallen iets
0170 ;fout. Dit komt omdat er geen rekening gehouden wordt met dit statement.
0180
0190 ;De omzetmethode hiervoor is gelijk aan de omzetregels voor GOTO en THEN,
0200 ;zodat het vrij simpel is om dit alsnog toe te voegen.
0210
0220 ;Ruud heeft hem destijds voor verschillende systemen opgezet welke alle
0230 ;MICROSOFT basic bezitten.
0240
0250 ;Indien dezelfde RUN faciliteit ook voor de andere systemen geldt, moet U
0260 ;zelf even het RUN token in Uw handboek opsnorren.
0270
0280 ;voor APPLE geldt
0290
0300 DATA .DE 131
0310 DIM .DE 134
0320 LET .DE 170
0330 GOTO .DE 171
0340 RUN .DE 172
0350 GOSUB .DE 176
0360 RETURN .DE 177
0370 REM .DE 178
0380 THEN .DE 196
0390
0400 ;Dezelfde tabel als op blz.11 6502 KENNER nr 25 toegevoegd met RUN.
0410
0420 ;Dan volgt nu de wijziging die nodig is in het gedeelte FURTHER
0430 ;terug te vinden op blz.17 van 6502 KENNER nr. 25
0440
0450
0460 .BA $91AE
0470
0480 FURTHER CMP #GOTO
0490 BEQ JUMP
0500 ;
0510 CMP #RUN ;nieuw!
0520 BEQ JUMP ;nieuw!
0530 ;
0540 CMP #GOSUB
0550 BEQ JMP
0560 ;enz enz enz
0570
0580
0590 ;Zij die hem in assembler hebben zullen hiermee geen problemen ondervinden,
0600 ;simpel intikken, en assembleren !!
0610
0620 ;alle hierna volgende opdrachten worden t.o.v. het originele programma wel
0630 ;4 plaatsen verschoven. Bij toekomstige veranderingen dient men hiermee
0640 ;terdege rekening te houden !!
0650
0660 ;Mooier is natuurlijk, ook voor hen die hem alleen nog maar in machinetaal
0670 ;hebben, om een "patch" aan te brengen.
0680
0690 ;hiertoe schrijven we eerst de volgende subroutine
0700
0710 .BA $8F00
0720
0730 STARTNIEUW JMP $9000

```

8F00- 4C 00 90

```

8F03- C9 AB 0740
8F05- F0 06 0750 RUNPATCH    CMP #GOTO
8F07- C9 AC 0760          BEQ RPEIND
8F09- F0 02 0770          CMP #RUN
8F0B- C9 C4 0780          BEQ RPEIND
8F0D- 60    0790          CMP #THEN
          0800 RPEIND      RTS
          0810
0820 ;Normaal zouden we zo'n subroutine na het programma schrijven, maar
0830 ;aangezien het programma gevolgd word door het werkgebied van de compresso
0840 ;(TABLE) zetten we hem dus simpel voor het programma.
0850
0860 ;De jump aan het begin van dit programma is om te verwijzen naar de
0870 ;originele start. Uw compressor start van nu af aan op $8F00.
0880
0890 ;Verder blijft er "enige" ruimte over voor toekomstige wijzigingen.
0900
0910 ; ($8FOE t/m $8FFF)
0920
0930 ;Dit omdat een programma onderhevig is aan evolutie.
0940
0950
0960 ;"RUNPATCH" zoekt alle statements op welke een gelijke uitwerking hebben.
0970
0980
0990 ;Nadat U dit verwerkt hebt zijn we toe aan de uitwerking van FURTHER(patch)
1000
1010          .BA $91AE
1020
1030
91AE- 20 03 BF 1040 FURTHER.    JSR RUNPATCH
91B1- F0 0F    1050          BEQ JUMP.
91B3- C9 B0    1060          CMP #GOSUB
91B5- F0 0D    1070          BEQ JMP.
91B7- EA      1080          NOP
91B8- EA      1090          NOP
91B9- EA      1100          NOP
          1110 ;
          1120 ;*****
          1130 ;
          1140          CMP #RETURN
          1150          BNE RPT
          1160          STA *TMPD
          1170          BEQ RPT
          1180
91C2- 85 09    1190 JUMP.      STA *TMPD
          1200
91C4- A0 01    1210 JMP.        LDY ##01
          1220 ;enz enz enz
          1230
          1240
          1250 ;WAT MOET U DOEN ???
          1260
          1270 ;om RUNPATCH dus te gebruiken laadt U eerst de originele versie.
          1280
          1290 ;Voor de duidelijkheid vullen we het gebied van $8F00-$8FFF natuurlijk
          1300 ;eerst met $00 (BRK) zodat we later weten welk gedeelte nog "vrij" is.
          1310
          1320 ;-U typt sprong en RUNPATCH vanaf $8F00 in.
          1330
          1340 ;n.b. voor andere computers:
          1350 ;      denk aan token + ander startadres.
          1360
          1370 ;-wijzigt dus FURTHER v.a. $91AE
          1380
          1390 ;-EN SAVE HET GEHEEL $8F00-$9300
          1400
          1410
          1420 ;Ruud Uphoff heeft bij zijn inleiding van de basiccompressor
          1430 ;6502 KENNER nr.21 blz.10 t/m 15
          1440 ;en bij de nabeschuiving
          1450 ;6502 KENNER nr.27 blz.12 t/m 14
          1460 ;nog eens aangegeven hoe en waarom de basic compressor te gebruiken.
          1470
          1480

```

```

1490 ;Ik zou hier het volgende aan toe willen voegen:
1500
1510 ;Bandbezitters hebben het voordeel allang door,
1520 ;zij zijn immers minder tijd kwijt voor laden en saven.
1530
1540 ;Terwijl disk gebruikers nu vaker, juist alle programma's die bij elkaar
1550 ;horen op 1 diskette weg kunnen zetten, door ze van te voren door de
1560 ;compressor te "wurmen".
1570
1580
1590 ; met groetjes van
1600
1610 ; Frans Verberkt
1620
1630
1640
1650
1660 TMPO .DE 9
1670 RPT .DE $9187
1680 JMP .DE $91C6
1690 JMP .DE $91C8
1700
1710 .EN

```

IPR#0  
LIST

```

10 REM *** DEMONSTRATIE PROGRAMMA VOOR HET MAKEN VAN GRAFIEKEN ***
15 REM *** HANS BOSCH
20 REM
25 REM *** DIT PROGRAMMA GEBRUIKT DE MACHINETAAL ROUTINE "GRAPHICS.OBJO"
30 REM *** ALS GEPUBLICEERD IN DE 6502 KENNER NR.41 VAN DECEMBER 1985, P.35 EV
35 REM
90 POKE 34,0: HOME :B$ = "EVEN GEDULD" ": PRINT B$
120 PRINT CHR$(4):"BRUN GRAPHICS.OBJ": REM $9059.$9485,$KARSET3 $9486.$95FF
125 IM = 200: DIM X(IM),Y(IM): HOME : GOTO 450
130 IM = 200
170 A = RND (1):E = 10 ^ (1 + 2 * INT (A - .5))
180 YL = 0:YH = YL:J = YL
190 I = I + 1:H = A * I
200 IF H - X(J) < A * 3 THEN 190
210 J = J + 1:A$ = ""
240 X(J) = H:H = A * E * 100 * SIN (H / 4) / (H + A)
250 IF YL > H THEN YL = H:A$ = "MIN": GOTO 280
260 IF YH < H THEN YH = H:A$ = "MAX"
280 Y(J) = H
290 IF I < IM THEN 190
300 IM = J:XH = X(J): HOME
310 REM ***** GRAPHICAL PART *****
320 HGR : SCALE= 1: ROT= 0: HCOLOR= 3
330 & HLIN 0,XH: & VLIN YL - .8 * YH,YH
340 & HLIN : & VLIN
350 FOR I = 1 TO IM
360 FOR J = 0 TO 4
370 & PLOT X(I),Y(I) - (J * YH) / 5,J
380 NEXT J,I
390 & DRAW "TEKST LANGS AS",X
400 & DRAW "RATE CM/S",Y
410 & DRAW "PROBE",210,20
420 FOR I = 0 TO 11: XDRAW 31: NEXT
430 ROT= 48: & DRAW "ROTATE",245,40
440 VTAB 22: PRINT "VERVOLG MET EEN TOETS": GET A$
450 VTAB 22: PRINT B$: GOTO 130

```

1



### FORTH on the VIC

Fred Behringer, Munchen, Germany.

I should like to get in contact with users of the DATATRONIC VIC 20 FORTH system. Newcomers welcome as I'm a novice too.

The first thing which was a source of annoyance to me when I started working with my VIC FORTH system was the incompleteness of the Reference Manual supplied by DATATRONIC: What about the missing description of the system's words KEY, PICK, UPPER, CURRENT, and FLUSH?

When first trying to get the whole list of FORTH words by typing in VLIST, I was surprised of how difficult it was to get hold of the information presented: things flushed by in no time.

So my first FORTH activity was to construct a new word, I called it \*LIST, which tries to circumvent the inconvenience mentioned. Here it is:

DECIMAL

: \*LIST

CR CR LATEST BEGIN DUP ID. CR PFA LFA @ KEY 13 = IF CR SP! ;S ENDIF DUP 0= UNTIL CR SP! ;

On being activated, \*LIST shows all words of the actual vocabulary, one word at a line, as long as any key (e.g. the CRSR key) is hit. As soon as the RETURN key is hit, the system returns to the terminal input loop.

A similar problem arose with DUMP. Here is my suggestion:

DECIMAL

: \*DUMP

BEGIN DUP DUP DUMP KEY 13 = IF ;S ENDIF 145 EMIT 4 + AGAIN ;

If called by ' n \*DUMP ', where n is the starting address, \*DUMP "dumps" as long as any key (e.g. the CRSR key) is hit. Return to the terminal by hitting (RETURN).

A third problem which came immediately to my mind was the problem of how to incorporate machine code programs into FORTH. Only think of the vast amount of ready made machine code subroutines contained in the 16K of the VIC's BASIC interpreter and its operating system! For instance, if I tried to change the tape operations initiated by CSAVE or CLOAD, I would be ill-advised if I did not use some of these handy routines. Now, the answer obviously is: Employ the ASSEMBLER. Unfortunately, the DATATRONIC VIC FORTH SYSTEM has no such thing. I am supposed instead to incorporate one by typing in a list of FORTH words supplied by DATATRONIC on an extra sheet within its Reference Manual. This seemed to me to be too much ado about almost nothing. Here, now, I was very pleased to come across the FORTH word:

HEX

6 USER SO

: JSR

HERE ! SP SO C! HERE ' LIT 17 + )R EXECUTE SP! 9E SO C! ;

which was recently published by Fridus Jonkman in DE 6502 KENNER (April 1985).

I tested JSR by typing in:

DECIMAL

160 828 C!	( LDY# )
5 829 C!	( 5 )
140 830 C!	( STY )
646 831 !	( \$0286 )
96 833 C!	( RTS )

And, alas, it did exactly what it was supposed to do: the TV screen colour was changed to green (actual colour held in address 646 by the VIC's operating system).

Of course, this could have been achieved faster than that by simply hitting or EMITting CTRL/GRN. But to me it was an impressive piece of demonstration of how easy it can be to incorporate machine code subroutines into FORTH.

\*\*\*\*\*  
 \*\*\*\*\* LET IT BE \*\*\*\*\*  
 \*\*\*\*\*

AUTEUR: Will Cuijpers, Tholen.

SYSTEM: Elektuur OCTOPUS DOS-COMPUTER

Over de muzikale capaciteiten van de CBM 64 is al genoeg lof geuit. Zoveel zelfs dat ik geïnteresseerd raakte in de hardware rond dit geluidsfenomeen. Uit de hardware van de CBM 64 en de Reference Guide begreep ik dat het een fluitje van een cent betekende: 32 vrije adresplaatsen, twee SN74LS688 en natuurlijk de 6581. De 6581 is verkrijgbaar voor prijzen die op dit moment sterk variëren. Van HFL 110,- bij DIL tot HFL 50,- in de USA. Deze laatste prijs is natuurlijk sterk afhankelijk van de dollarkoers.

Adressering van de 6581 lijkt mij voor ieder duidelijk. Bij mij in een OCTOPUS staat de 6581 op \$E160 tot \$E17F. Dus A15, A14, A13, A8, A6 en A5 staan met de DIL-schakelaars open, terwijl de A12, A11, A10, A9 en A7 via de DIL-schakelaars op staan. Men diene er wel op te letten dat de A4 adreslijn op de chip zit, dus een bereik van X150-X16F gaat niet. Voor verdere informatie verwijst ik naar de Reference Guide de CBM 64 en natuurlijk alle voorbeeldprogramma's die bestaan.

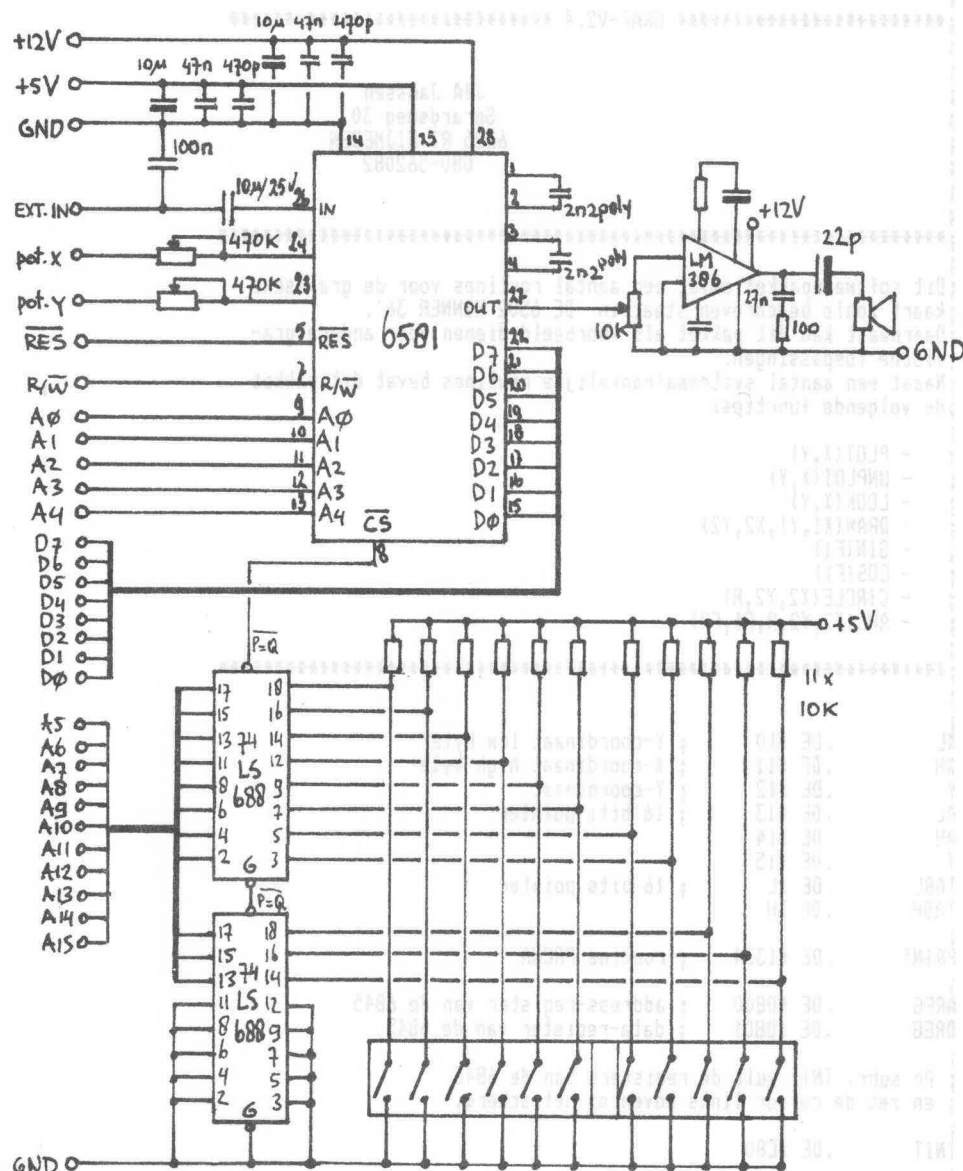
```
1 PRINTCHR$(12):PRINT"
4 FOR I=57728TO57752
5 POKE I,0
6 NEXT I
10 EERSTE=57728
11 VOLUME=EERSTE+24
13 UIT=EERSTE+6
14 H1=EERSTE
15 H2=EERSTE+1
16 GATE=EERSTE+4
20 POKEVOLUME,15
21 POKEAAN,23
22 POKEUIT,123
30 READ HOOGTE,DUUR
40 IF HOOGTE=0THENEND
50 F2=HOOGTE/256:F1=HOOGTE-F2*256
```

\*\*\*\*\* LET IT BE \*\*\*\*\*

```
60 POKEH2,F2:POKEH1,F1
70 POKEGATE,33:FORI=0TODUUR*100
75 NEXT
80 POKEGATE,32:FORI=0TODUUR*100
85 NEXT
90 GOTO30
100 DATA6430,1,6430,1,6430,3,6430,1,7217,2
150 DATA5407,2,6430,1,6430,2,8583,3,9634,2
200 DATA9634,1,10814,2,10814,3,9634,2,9634,2
250 DATA8583,1,8583,5,10814,1,10814,2,11457,3,10820,2
300 DATA10814,2,9634,4,10814,1,9634,1,9634,1,8583,11
400 DATA10814,1,9634,2,8534,5,10814,1,12860,2,14435,5
450 DATA12860,1,12860,2,10814,3,6439,2,6439,1,6439,2
500 DATA10814,9,10814,1,10814,2,11457,3,10820,2,10814,2
550 DATA9634,4,10814,1,9634,1,9634,1,8583,15,0,0
(zie voor hardware-schema volgende pagina)
```

De redactie roept hierbij alle bezitters van EC65 en OCTOPUS op hun zelf ontwikkelde software op te sturen naar de redactie en zo mee te helpen aan de verdere uitwisseling van kennis en ervaringen.

Van Marcel Visser vernamen wij het volgende. Het aanschakelen van zijn 80-koloms kaart (AP16 van IBS Computertechniek) op de APPLE II gaat met PR#3, het uitschakelen door het intypen van CTRL-Z1. De reden van deze vreemde control code is: indien de kaart wordt aangeschakeld, wordt zowel de output-hook als de input-hook veranderd. Het aanschakelen gebeurt in Basic (eigenlijk DOS) door een JSR \$CX00. Wanneer nu de kaart wordt uitgeschakeld met PR#0, dan zet Basic de output-hook op COUT1 (\$FDF0). De input-hook staat nog steeds op de 80-kolomskaart en de videoswitch staat nog steeds op de 80-kolomspagina (de videoswitch is een softswitch vergelijkbaar met de softswitches van de Hires- en Lowres-pagina's). Je komt na het commando PR#0 dus in een soort niemandsland, waarbij tekst geschreven wordt naar het 40-kolomsscherm, terwijl je het 80-kolomsscherm ziet. De APPLE IIe 80-kolomskaart heeft iets soortgelijks. Hier gebeurt het uitschakelen met een escape-code.  
 (Red.: De redactie verneemt graag wie dit probleem herkent. Zij verneemt nog liever hoe dit op te lossen is.)



Aan de tekentafel:  
Fridus Jonkman

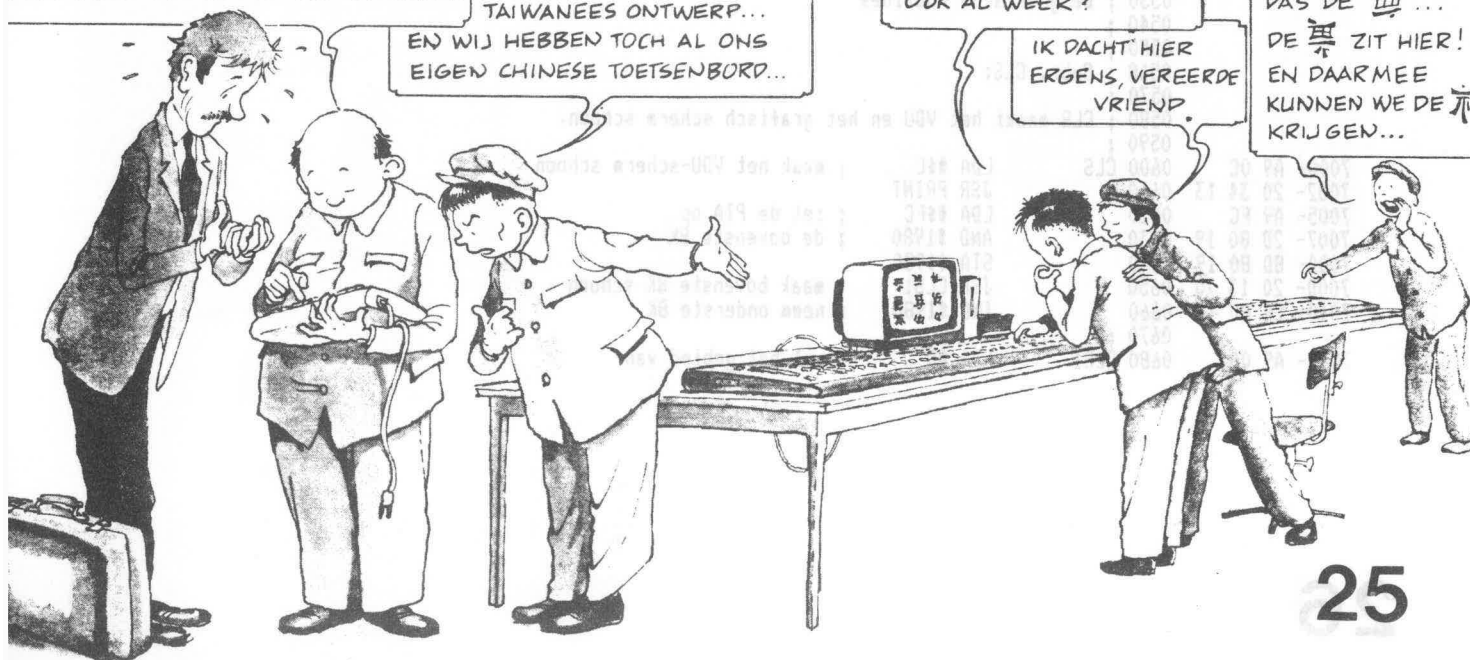
BLECHTS 26 TOETSEN? MAAR NEE,  
VERBODWAARDIGE, WIJ HEBBEN  
TOCH VELE DUIZENDEN LETTERTEKENS

EN DAN... HET IS EEN  
TAIWANEES ONTWERP...  
EN WIJ HEBBEN TOCH AL ONS  
EIGEN CHINESE TOETSENBOORD...

EENS EVEN KIJKEN...  
WAAR ZIT DE 𠄎  
OOK AL WEER?

IK DACHT HIER  
ERGENS, VEREERDE  
VRIEND.

NEE, KAMERADEN  
DA'S DE 𠄎...  
DE 𠄎 ZIT HIER!  
EN DAARMEE  
KUNNEN WE DE 𠄎  
KRIJGEN...





0010 :  
0020 :\*\*\*\*\* GRAF-V2.4 \*\*\*\*\*

JJA Janssen  
Gerardsweg 30  
6525 RT NIJMEGEN  
080-562082

0100 :\*\*\*\*\*

0110 :  
0120 :Dit softwarepakket omvat een aantal routines voor de grafische  
0130 :kaart zoals beschreven staat in 'DE 6502 KENNER 36'.

0140 :Daarnaast kan dit pakket als voorbeeld dienen voor andere gra-  
0150 :fische toepassingen.

0160 :Naast een aantal systeemafhankelijke routines bevat dit pakket  
0170 :de volgende functies:

0180 :  
0190 : - PLOT(X,Y)  
0200 : - UNPLOT(X,Y)  
0210 : - LOOK(X,Y)  
0220 : - DRAW(X1,Y1,X2,Y2)  
0230 : - SIN(F1)  
0240 : - COS(F1)  
0250 : - CIRCLE(X2,Y2,R)  
0260 : - ARC(X2,Y2,R,F1,F2)

0270 :  
0280 :\*\*\*\*\*

0290 :  
0300 :  
0305 XL .DE \$10 ; Y-coördinaat low byte  
0310 XH .DE \$11 ; X-coördinaat high byte  
0320 Y .DE \$12 ; Y-coördinaat  
0330 AL .DE \$13 ; 16 bits pointer  
0340 AH .DE \$14  
0350 T .DE \$15  
0360 TABL .DE XL ; 16 bits pointer  
0370 TABH .DE XH

0380 :  
0390 PRINT .DE \$1334 ; routine PRCHA

0400 :  
0410 AREG .DE \$D800 ; address-register van de 6845  
0420 DREG .DE \$D801 ; data-register van de 6845

0430 :  
0440 : De subr. INIT vult de registers van de 6845  
0450 : en zet de cursor links boven op het scherm.

0460 :  
0470 INIT .DE \$C80

0480 :  
0490 : .BA \$7000  
0500 : .OS

0510 :  
0520 :  
0530 : De grafische routines

0540 :  
0550 :  
0560 : Subr. CLS:

0570 :  
0580 : CLS maakt het VDU en het grafisch scherm schoon.

0590 :  
0600 CLS LDA #\$C ; maak het VDU-scherm schoon

0610 JSR PRINT

0620 LDA #\$FC ; zet de PIA op

0630 AND \$1980 ; de bovenste 8K

0640 STA \$1980

0650 JSR CLS1 ; maak bovenste 8K schoon

0660 INC \$1980 ; neem onderste 8K

0670 :

7013- A9 00 0680 CLS1 LDA #0 ; vul het gebied van

```

7015- A8      0690      TAY          ; $B000 tot $D000 met $00
7016- 85 13   0700      STA #AL
7018- A9 B0   0710      LDA #B0      ; ADRES := $B000
701A- 85 14   0720      STA #AH
701C- A9 00   0730      LDA #00      ; REPEAT
701E- 2C 82 19 0740      CLS2
CLS3      BIT $1982
7021- 30 FB   0750      BMI CLS3      ; wacht op sync.
7023- 91 13   0760      STA (AL),Y    ; (ADRES,Y) := 0
7025- C8      0770      INY
7026- D0 F6   0780      BNE CLS3
7028- E6 14   0790      INC #AH      ; ADRES,Y := ADRES,Y + 1
702A- A5 14   0800      LDA #AH
702C- C9 D0   0810      CMP #D0      ; UNTIL ADRES = $D000
702E- D0 EC   0820      BNE CLS2
          0830      ;
7030- A9 0C   0840      LDA #C
7032- 20 34 13 0850      JSR PRINT    ; maak VDU-scherm schoon
7035- 60      0860      RTS
          0870      ;
          0880      ;
          0890      ; Subr. PLOT zet een puntje weg op de coördinaten X,Y
          0900      ; dit gebeurt gedurende de synchronisatie, zodat er geen
          0910      ; storingen op het beeld komen, waardoor de snelheid
          0920      ; afneemt. Indien dit niet gewenst is kunnen de loops
          0930      ; PW1, PW2, UW1, UW2 e.t.c. vervangen worden door 5x NOP.
          0940      ;
          0950      ; INPUT : XH,XL X-coördinaat
          0960      ; Y Y-coördinaat
          0970      ; OUTPUT: een puntje op het scherm op de plaats X,Y
          0980      ;
7036- 20 67 70 0990      PLOT      JSR PLOT1    ; bereken adres en bitpatroon
7039- 2C 82 19 1000      PW1      BIT $1982    ; wacht
703C- 30 FB   1010      BMI PW1      ; op sync
703E- 11 13   1020      ORA (AL),Y
7040- 2C 82 19 1030      PW2      BIT $1982    ; wacht
7043- 30 FB   1040      BMI PW2      ; op sync
7045- 91 13   1050      STA (AL),Y    ; set het bitje
7047- 60      1060      RTS
          1070      ;
          1080      ; Subr. UNPLOT is gelijk aan subr. PLOT, alleen nu wordt het
          1090      ; puntje weggehaald.
          1100      ;
7048- 20 67 70 1110      UNPLOT    JSR PLOT1
704B- 49 FF   1120      EOR #FF
704D- 2C 82 19 1130      UW1      BIT $1982
7050- 30 FB   1140      BMI UW1
7052- 31 13   1150      AND (AL),Y
7054- 2C 82 19 1160      UW2      BIT $1982
7057- 30 FB   1170      BMI UW2
7059- 91 13   1180      STA (AL),Y    ; reset het bitje
705B- 60      1190      RTS
          1200      ;
          1210      ; Subr. LOOK kijkt of er op de plaats X,Y een puntje staat
          1220      ;
          1230      ; INPUT : XH,XL X-coördinaat
          1240      ; Y Y-coördinaat
          1250      ; OUTPUT: ACCU ACCU = 0 (Z-flag = 1) ==> puntje niet aanwezig
          1260      ; ACCU > 0 (Z-flag = 0) ==> puntje wel aanwezig
          1270      ;
705C- 20 67 70 1280      LOOK      JSR PLOT1    ; bereken adres en bitpatroon
705F- 2C 82 19 1290      LW2      BIT $1982    ; wacht
7062- 30 FB   1300      BMI LW2      ; op sync
7064- 31 13   1310      AND (AL),Y    ; zet ACCU en Z-flag goed
7066- 60      1320      RTS
          1330      ;
          1340      ; .CT

```

F02 101C 41F0-520C

```

1040      ;
1050      ; PLOT1 berekent uit de X,Y-coördinaten de geheugenplaats

```

```

1060 ; en bitpatroon ($80, $40 ... $02, $01) van het puntje.
1070
1080 INPUT : XH,XL 10 bits X-coördinaat (0-640)
1090 Y 8 bits Y-coördinaat (0-200)
1100 OUTPUT: AH,AL adres van de coördinaten X,Y
1110 ACCU bitpatroon
1120
1130 ; Het adres wordt berekend met de formule:
1140
1150 AH,AL := (8*INT(Y/8)*80 + 8*INT(X/8) + (Y-8*INT(Y/8) + $B000
1160
1170 ; Het bitnummer volgt uit de formule:
1180
1190 ACCU := X-8*INT(X/8)
1200
1210 ; Met dit bitnummer wordt, d.m.v. de tabel PLOTTAB, het
1220 ; desbetreffende bitje geset.
1230

```

```

7067- A9 00 1240 PLOT1 LDA #0
7069- A8 1250 TAY
706A- B5 14 1260 STA *AH
706C- A5 12 1270 LDA *Y
706E- 29 07 1280 AND #$7
7070- B5 15 1290 STA *T ; T := Y-8*INT(Y/8)
7072- A5 12 1300 LDA *Y
7074- 29 F8 1310 AND #$FB
7076- B5 13 1320 STA *AL ; AH,AL := 8*INT(Y/8)
7078- 06 13 1330 ASL *AL
707A- 26 14 1340 ROL *AH
707C- 06 13 1350 ASL *AL
707E- 26 14 1360 ROL *AH ; AH,AL := 8*INT(Y/8)*4
7080- 18 1370 CLC
7081- 65 13 1380 ADC *AL
7083- B5 13 1390 STA *AL
7085- A5 14 1400 LDA *AH
7087- 69 00 1410 ADC #0
7089- B5 14 1420 STA *AH ; AH,AL := 8*INT(Y/8)*5
708B- 06 13 1430 ASL *AL
708D- 26 14 1440 ROL *AH
708F- 06 13 1450 ASL *AL
7091- 26 14 1460 ROL *AH
7093- 06 13 1470 ASL *AL
7095- 26 14 1480 ROL *AH
7097- 06 13 1490 ASL *AL
7099- 26 14 1500 ROL *AH ; AH,AL := 8*INT(Y/8)*80
709B- A5 10 1510 LDA *XL
709D- 29 F8 1520 AND #$FB
709F- 05 15 1530 ORA *T ; AH,AL := 8*INT(Y/8)*80 +
70A1- 18 1540 CLC ; 8*INT(X/8) +
70A2- 65 13 1550 ADC *AL ; Y-8*INT(Y/8) +
70A4- B5 13 1560 STA *AL ; $B000
70A6- A5 14 1570 LDA *AH
70A8- 69 B0 1580 ADC #$B0
70AA- 65 11 1590 ADC *XH
70AC- 09 B0 1600 ORA #$B0 ; AH,AL > $8000 (voor de veiligheid)
70AE- B5 14 1610 STA *AH
70B0- C9 D0 1620 CMP #$D0 ; 1ste of 2de blok ?
70B2- 30 0E 1630 BMI NOHIGH
70B4- 38 1640 SEC ;
70B5- E9 20 1650 SBC #$20 ; 2de blok
70B7- B5 14 1660 STA *AH ; zet de PIA goed
70B9- AD B0 19 1670 LDA $19B0
70BC- 29 FD 1680 AND #$FD
70BE- 09 01 1690 ORA #$01
70C0- D0 05 1700 BNE PLOT1
70C2- AD B0 19 1710 NOHIGH LDA $19B0 ; 1ste blok
70C5- 29 FC 1720 AND #$FC ; zet de PIA goed
70C7- BD B0 19 1730 PLOT1 STA $19B0
70CA- A5 10 1740 LDA *XL
70CC- 29 07 1750 AND #$7 ; ACCU := bitnummer
70CE- AA 1760 TAX
70CF- BD D3 70 1770 LDA PLOTTAB,X

```



```

70D2- 60      1780      RTS      ; ACCU := bitpatroon
              1790 ;
70D3- 80 40 20 1800 PLOTTAB .BY $80 $40 $20 $10 8 4 2 1
70D6- 10 08 04
70D9- 02 01

1810 ;
1820 ;
1830 ; Subr. INITPIA initialiseert de PIA (6821), na een
1840 ; reset dient deze routine aangeroepen te worden.
1850 ;
70DB- A9 00      1860 INITPIA LDA #00
70DD- BD 81 19   1870 STA $1981 ; CRA-2 := 0
70E0- BD 83 19   1880 STA $1983 ; CRB-2 := 0
70E3- BD 82 19   1890 STA $1982 ; poort B := input
70E6- A9 FF      1900 LDA #$FF
70EB- BD 80 19   1910 STA $1980 ; poort A := output
70EB- A9 04      1920 LDA #4
70ED- BD 81 19   1930 STA $1981 ; CRA-2 := 1
70F0- BD 83 19   1940 STA $1983 ; CRB-2 := 1
70F3- A9 1C      1950 LDA #$1C
70F5- BD 80 19   1960 STA $1980 ; grafisch display uit
70F8- 60      1970 RTS
              1980 ;
              1990 ;
2000 ; Subr. ON zet het grafisch display aan en vult de
2010 ; registers van de 6845 met de waarden uit de tabel
2020 ; TAB, zodat er een trillingsvrij beeld ontstaat.
2030 ;
2040 ; Het display komt in de niet geïnverteerde
2050 ; twee kleuren mode.
2060 ;
70F9- A9 1C      2070 ON LDA #$1C
70FB- 20 34 13   2080 JSR PRINT ; cursor home
70FE- A9 F3      2090 LDA #$F3
7100- 2D 80 19   2100 AND $1980 ; Pa2 en Pa3 := 0 ==>
7103- 09 10      2110 ORA #$10 ; 2 kleuren
7105- BD 80 19   2120 STA $1980 ; Pa4 := 1 ==> normaal beeld
7108- A2 09      2130 LDX #9 ; X := 9
710A- BE 00 DB   2140 ON1 STX AREG ; REPEAT
7110- BD 17 71   2150 LDA TAB,X ; address register := X
7110- BD 01 DB   2160 STA DREG ; data register := TAB(X)
7113- CA      2170 DEX ; X := X - 1
7114- 10 F4      2180 BPL ON1 ; UNTIL X < 0
7116- 60      2190 RTS
              2200 ;
              2210 ; Waarden voor de registers van de 6845 om een
              2220 ; stilstaand beeld te krijgen, deze moeten misschien
              2230 ; aangepast worden (m.u.v. 2e, 7e, 9e en 10e byte).
              2240 ;
7117- 78 50 5C   2250 TAB .BY $78 $50 $5C $0A $26
711A- 0A 26
711C- 00 19 1F   2260 .BY $00 $19 $1F $00 $07
711F- 00 07
              2270 ;
              2280 ; Subr. OFF zet het display uit.
              2290 ;
7121- A9 1C      2300 OFF LDA #$1C
7123- BD 80 19   2310 ORA $1980
7126- BD 80 19   2320 STA $1980 ; Pa2, Pa3, Pa4 = 1
7129- 4C 80 0C   2330 JMP INIT ; initialiseer VDU-scherm
              2340 ;
              2350 ; Subr. INV invertteert het grafisch display.
              2360 ;
712C- A9 10      2370 INV LDA #$10
712E- 4D 80 19   2380 EOR $1980 ; toggle PA4
7131- BD 80 19   2390 STA $1980
7134- 60      2400 RTS
              2410 ;
              2420 ;
2430 ; DRIEKL zet het grafisch display in de drie kleuren-
2440 ; mode. De horizontale resolutie neemt een factor 2 af.
2450 ; De knipperfunctie is nog steeds te gebruiken.

```

```

2460 ;
7135- AD 80 19 2470 DRIEKL LDA #1980
7138- 29 F7 2480 AND #F7
713A- 09 04 2490 ORA #04
713C- 8D 80 19 2500 STA #1980
713F- 60 2510 RTS
2520 ;
2530 ; ACHTKL zet het grafisch diplay in de acht kleurenmode.
2540 ; De horizontale resolutie is een factor 2 kleiner dan
2550 ; bij DRIEKL. De knipperfunctie is niet meer te gebruiken.
2560 ;
7140- AD 80 19 2570 ACHTKL LDA #1980
7143- 29 FB 2580 AND #FB
7145- 09 08 2590 ORA #08
7147- 8D 80 19 2600 STA #1980
714A- 60 2610 RTS
2620 ;
2630 ; KNIPON zet de knippermode aan, het knipperen
2640 ; gebeurt softwarematig met de routine TOGGLE.
2650 ; De horizontale resolutie wordt een factor 2 kleiner.
2660 ; Voor hardwarematig knipperen moet gelden Pa6=1.
2670 ; (zie ook de tabel in 'DE 6502 KENNER 36' blz. 6).
2680 ;
714B- AD 80 19 2690 KNIPON LDA #1980
714E- 29 BF 2700 AND #BF
7150- 09 20 2710 ORA #20
7152- 8D 80 19 2720 STA #1980
7155- 60 2730 RTS
2740 ;
2750 ; KNIPOFF zet de knippermode uit.
2760 ; De horizontale resolutie wordt weer een factor 2 groter.
2770 ;
7156- AD 80 19 2780 KNIPOFF LDA #1980
7159- 29 DF 2790 AND #DF
715B- 09 04 2800 ORA #04
715D- 8D 80 19 2810 STA #1980
7160- 60 2820 RTS
2830 ;
2840 ; Na elke aanroep van TOGGLE wordt het MSB-bitje van elk punt
2850 ; geïnverteerd.
2860 ; Indien MSB=1 --> punt aanwezig.
2870 ; Indien MSB=0 --> punt niet aanwezig.
2880 ;
7161- AD 80 19 2890 TOGGLE LDA #1980
7164- 49 80 2900 EOR #80
7166- 8D 80 19 2910 STA #1980
7169- 60 2920 RTS
2930 ;
2940 ; .CT

```

F03 1882 41F0-5A72

```

0010 ;
0020 ; Parameters voor de routine DRAW.
0030 ;
0040 X1L .DE XL
0050 X1H .DE XH ; X1,Y1 is beginpunt v/d lijn
0060 Y1 .DE Y
0070 X2L .DE #16
0080 X2H .DE #17 ; X2,Y2 is eindpunt v/d lijn
0090 Y2 .DE #18
0100 DXL .DE #19
0110 DXH .DE #1A ; DXH,DXL := ABS(X2-X1)
0120 DY .DE #1B ; DY := ABS(Y2-Y1)
0130 S .DE #1C ; stapgrootte
0140 R .DE #1D ; restwaarde v/d stapgrootte
0150 TEMP .DE #1E ; scratch geheugenplaats
0160 ;
0170 ; Parameters voor de routine DIVIDE.
0180 ;
0190 DIVI .DI DY ; deeltal high byte

```

```

0200 DIVIL .DE $1F ; deeltal low byte
0210 DIVR .DI DXH ; deler high byte
0220 DIVRL .DI DXL ; deler low byte
0230 DIVD .DI S ; quotient high byte
0240 DIVOL .DI R ; quotient low byte
0250 DIVSC .DI AL ; scratch geheugen
0260 DIVSCL .DI AH ; idem
0270 ;
0271 ; DIVIDE voert de volgende deling uit:
0370 ;
0380 ; (DIVD,DIVOL) := (DIVI,DIVIL) / (DIVR,DIVRL)
0390 ; <-H->,<--L-->
0400 ;
0401 ;
0402 ; In de routine DIVIDE wordt de stapgrootte S (met fractie R) uitgerekend.
0403 ;
0404 ; S,R := 256*DY / DX
0405 ;
716A- A0 10 0410 DIVIDE LDY #16
716C- A9 00 0420 LDA #0
716E- B5 13 0430 STA *DIVSC
7170- B5 14 0440 STA *DIVSCL
7172- 06 1F 0450 DIVI ASL *DIVIL ; Herhaal voor 16 bits
7174- 26 1B 0460 ROL *DIVI ; Schuif DIVI
7176- 26 14 0470 ROL *DIVSCL
7178- 26 13 0480 ROL *DIVSC ; Rol carry in DIVSC
717A- 38 0490 SEC
717B- A5 14 0500 LDA *DIVSCL ; IF DIVSC >= DIVR THEN
717D- E5 19 0510 SBC *DIVRL
717F- B5 1E 0520 STA *TEMP ; BEGIN
7181- A5 13 0530 LDA *DIVSC
7183- E5 1A 0540 SBC *DIVR ; DIVSC := DIVSC - DIVR
7185- 90 06 0550 BCC DIV2 ; carry := 1
7187- B5 13 0560 STA *DIVSC
7189- A5 1E 0570 LDA *TEMP ; END
718B- B5 14 0580 STA *DIVSCL ; ELSE carry := 0
718D- 26 1D 0590 ;
718F- 26 1C 0600 DIV2 ROL *DIVOL ; rol carry in DIVO
7191- B8 0620 DEY
7192- D0 DE 0630 BNE DIV1 ; Neem het volgende bit
7194- 60 0640 RTS
0650 ;
0660 ; Subr. DX>DY kijkt of DX groter is dan DY.
0665 ;
0670 ; INPUT : DXH,DXL two's complement van DX
0680 ; DY two's complement van DY
0690 ; OUTPUT: N-flag N=0 indien DX>DY
0695 ;
7195- A5 1A 0700 DX>DY LDA *DXH
7197- D0 0B 0710 BNE DX1
7199- A5 19 0720 LDA *DXL
719B- 45 1B 0730 EOR *DY
719D- 30 06 0740 BMI DX2
719F- 38 0750 SEC
71A0- A5 19 0760 LDA *DXL
71A2- E5 1B 0770 SBC *DY
71A4- 60 0780 DX1 RTS
71A5- A5 1B 0790 DX2 LDA *DY
71A7- 60 0800 RTS
0810 ;
0815 ; CHANGE verwisselt begin- en eindpunt v/e lijn.
0816 ; Behoort bij routine DRAW.
0817 ;
71A8- A5 10 0820 CHANGE LDA *X1L
71AA- 4B 0830 PHA
71AB- A5 11 0840 LDA *X1H
71AD- 4B 0850 PHA
71AE- A5 12 0860 LDA *Y1
71B0- 4B 0870 PHA
71B1- A5 1B 0880 LDA *Y2
71B3- B5 12 0890 STA *Y1

```



```

71B5- A5 16 0900 LDA *X2L
71B7- B5 10 0910 STA *X1L
71B9- A5 17 0920 LDA *X2H
71BB- B5 11 0930 STA *X1H
71BD- B8 0940 PLA
71BE- B5 18 0950 STA *Y2
71C0- B8 0960 PLA
71C1- B5 17 0970 STA *X2H
71C3- B8 0980 PLA
71C4- B5 16 0990 STA *X2L
1000 ;
1001 ; Subr. DRAW trekt een lijn tussen twee punten.
1002 ;
1003 ; INPUT : X1H,X1L X-coördinaat beginpunt (0-640)
1004 ; Y1 Y-coördinaat beginpunt (0-200)
1005 ; X2H,X2L X-coördinaat eindpunt (0-640)
1006 ; Y2 Y-coördinaat eindpunt (0-200)
1007 ;
71C6- B8 1010 DRAW SEC
71C7- A5 16 1020 LDA *X2L
71C9- E5 10 1030 SBC *X1L
71CB- B5 19 1040 STA *DXL ; DX := X2 - X1
71CD- A5 17 1050 LDA *X2H
71CF- E5 11 1060 SBC *X1H
71D1- B5 1A 1070 STA *DXH
71D3- B0 D3 1080 BMI CHANGE ; verwissel begin- eindpunt indien DX<0
1081 ;
1082 ; bereken de richting v/d lijn.
1083 ;
71D5- A5 12 1090 LDA *Y1
71D7- A5 18 1100 EOR *Y2
71D9- B0 0B 1110 BMI DRAW1
71DB- B8 1120 SEC
71DC- A5 18 1130 LDA *Y2
71DE- E5 12 1140 SBC *Y1
71E0- F0 0E 1150 BEQ HOR
71E2- B0 09 1160 BPL SITD
71E4- D0 04 1170 BNE SITA
71E6- A5 12 1180 DRAW1 LDA *Y1
71E8- B0 03 1190 BPL SITD
71EA- 4C F3 71 1200 SITA JMP DSITA ; Dalende lijn
71ED- 4C B8 72 1210 SITD JMP DSITD ; Stijgende lijn
71F0- 4C E7 72 1220 HOR JMP DHOR ; Horizontale lijn
1230 ;
1240 ; RC v/d lijn 0-90 graden
1250 ;
71F3- B8 1260 DSITA SEC
71F4- A5 12 1270 LDA *Y1
71F6- E5 18 1280 SBC *Y2 ; DY := Y1 - Y2
71F8- B5 18 1290 STA *DY
71FA- B0 95 71 1300 JSR DX>DY ; Indien DX<DY dan
71FD- B0 23 1310 BMI DS1 ; RC v/d lijn is 0-45 graden
1311 ;
1312 ; RC v/d lijn is 45-90 graden.
1313 ;
71FF- B0 6A 71 1320 JSR DIVIDE ; Bereken stapgrootte
7202- A9 00 1330 LDA #0
7204- B5 1E 1340 STA *TEMP
7206- B0 B3 72 1350 DS2 JSR PRNDY ; REPEAT
7209- B8 1360 SEC
720A- A5 1E 1370 LDA *TEMP ; PLOT(X1,Y1)
720C- E5 1D 1380 SBC *R
720E- B5 1E 1390 STA *TEMP ; Y1 := Y1 - stapgrootte
7210- A5 12 1400 LDA *Y1
7212- E5 1C 1410 SBC *S ; X1 := X1 + 1
7214- B5 12 1420 STA *Y1
7216- B0 93 72 1430 JSR INCX1
7219- D0 EB 1440 BNE DS2 ; UNTIL X1=X2
721B- A5 18 1450 LDA *Y2
721D- B5 12 1460 STA *Y1
721F- 4C B6 70 1470 JMP PLOT ; PLOT(X2,Y2)
1480 ;

```

```

1490 ; RC v/d lijn 0-45 graden
1500 ;
7222- 20 D5 72 1510 DS1 JSR CHDXDY ; Verwissel DX en DY en bereken stapgrootte
7225- 20 A4 72 1520 DS4 JSR PADDX1 ; REPEAT
7228- C6 12 1530 DEC #Y1 ; PLOT(X1,Y1)
722A- A5 12 1540 LDA #Y1 ; X1 := X1 + stapgrootte
722C- C5 18 1550 CMP #Y2 ; Y1 := Y1 - 1
722E- D0 F5 1560 BNE DS4 ; UNTIL Y1=Y2
7230- A5 17 1570 LDA #X2H
7232- B5 11 1580 STA #X1H
7234- A5 16 1590 LDA #X2L
7236- B5 10 1600 STA #X1L
7238- 4C 36 70 1610 JMP PLOT ; PLOT(X2,Y2)
1620 ;
1630 ; RC v/d lijn 270-360 graden
1640 ;
723B- 38 1650 DS1TD SEC
723C- A5 18 1660 LDA #Y2
723E- E5 12 1670 SBC #Y1 ; DY := Y2 - Y1
7240- B5 18 1680 STA #DY
7242- 20 95 71 1690 JSR DX>DY ; Indien DX<DY dan
7245- 30 23 1700 BMI DS8 ; RC v/d lijn is 270-315 graden
1701 ;
1702 ; RC v/d lijn is 315-360 graden
1703 ;
7247- 20 6A 71 1710 JSR DIVIDE ; Bereken stapgrootte
724A- A9 00 1720 LDA #0
724C- B5 1E 1730 STA #TEMP
724E- 20 83 72 1740 DS5 JSR PRNDY ; REPEAT
7251- 18 1750 CLC
7252- A5 1E 1760 LDA #TEMP ; PLOT(X1,Y1)
7254- 65 1D 1770 ADC #R
7256- B5 1E 1780 STA #TEMP ; Y1 := Y1 + stapgrootte
7258- A5 12 1790 LDA #Y1
725A- 65 1C 1800 ADC #S ; X1 := X1 + 1
725C- B5 12 1810 STA #Y1
725E- 20 93 72 1820 JSR INCX1
7261- D0 EB 1830 BNE DS5 ; UNTIL X1=X2
7263- A5 18 1840 LDA #Y2
7265- B5 12 1850 STA #Y1
7267- 4C 36 70 1860 JMP PLOT ; PLOT(X2,Y2)
1870 ;
1880 ; RC v/d lijn 270-315 graden
1890 ;
726A- 20 D5 72 1900 DS8 JSR CHDXDY ; Verwissel DX en DY en bereken stapgrootte
726D- 20 A4 72 1910 DS7 JSR PADDX1 ; REPEAT
7270- E6 12 1920 INC #Y1 ; PLOT(X1,Y1)
7272- A5 12 1930 LDA #Y1 ; X1 := X1 + stapgrootte
7274- C5 18 1940 CMP #Y2 ; Y1 := Y1 + 1
7276- D0 F5 1950 BNE DS7 ; UNTIL Y1=Y2
7278- A5 17 1960 LDA #X2H
727A- B5 11 1970 STA #X1H
727C- A5 16 1980 LDA #X2L
727E- B5 10 1990 STA #X1L
7280- 4C 36 70 2000 JMP PLOT ; PLOT(X2,Y2)
2010 ;
2015 ;PRNDY rondt Y1 af (fractie van Y1 staat in TEMP), waarna de
2016 ;routine PLOT wordt aangeroepen.
2017 ;Bij het verlaten v/d routine PRNDY bezit Y1 weer de oude waarde.
2018 ;
7283- A5 1E 2020 PRNDY LDA #TEMP
7285- 10 02 2030 BPL PR1
7287- E6 12 2040 INC #Y1
7289- 20 36 70 2050 PR1 JSR PLOT
728C- A5 1E 2060 LDA #TEMP
728E- 10 02 2070 BPL PRE
7290- C6 12 2080 DEC #Y1
7292- 60 2090 PRE RTS
2100 ;
2101 ; INCX1:
2102 ; X1 := X1 + 1
2103 ; Z-flag := 1 indien X1=X2

```

```

2104 ;
7293- E6 10 2110 INCX1 INC *X1L
7295- D0 02 2120 BNE CMPX
7297- E6 11 2130 INC *X1H
7299- A5 10 2140 CMPX LDA *X1L
729B- C5 16 2150 CMP *X2L
729D- D0 04 2160 BNE INCXE
729F- A5 11 2170 LDA *X1H
72A1- C5 17 2180 CMP *X2H
72A3- 60 2190 INCXE RTS
2200 ;
2201 ;PADDX1 rondt X1 af (fractie van X1 staat in TEMP) en roept
2202 ;de routine PLOT aan, hierna wordt bij de oude waarde van X1
2203 ;de stapgrootte (S met fractie R) opgeteld.
2204 ;
72A4- A5 1E 2210 PADDX1 LDA *TEMP
72A6- 10 0C 2220 BPL PA
72A8- A5 10 2230 LDA *X1L
72AA- 48 2240 PHA
72AB- A5 11 2250 LDA *X1H
72AD- 48 2260 PHA
72AE- E6 10 2270 INC *X1L
72B0- D0 02 2280 BNE PA
72B2- E6 11 2290 INC *X1H
72B4- 20 36 70 2300 PA JSR PLOT
72B7- A5 1E 2310 LDA *TEMP
72B9- 10 06 2320 BPL PA1
72BB- 68 2330 PLA
72BC- 85 11 2340 STA *X1H
72BE- 68 2350 PLA
72BF- 85 10 2360 STA *X1L
2370 PA1
72C1- 18 2380 CLC
72C2- A5 1E 2390 LDA *TEMP
72C4- 65 1D 2400 ADC *R
72C6- 85 1E 2410 STA *TEMP
72C8- A5 10 2420 LDA *X1L
72CA- 65 1C 2430 ADC *S
72CC- 85 10 2440 STA *X1L
72CE- A5 11 2450 LDA *X1H
72D0- 69 00 2460 ADC #0
72D2- 85 11 2470 STA *X1H
72D4- 60 2480 RTS
2490 ;
2491 ;CHDXDY verwisselt DX en DY en berekent de stapgrootte.
2492 ;
72D5- A5 19 2500 CHDXDY LDA *DXL
72D7- 48 2510 PHA
72D8- A5 1B 2520 LDA *DY
72DA- 85 19 2530 STA *DXL
72DC- 68 2540 PLA
72DD- 85 1B 2550 STA *DY
72DF- 20 6A 71 2560 JSR DIVIDE ; Bereken stapgrootte
72E2- A9 00 2570 LDA #0
72E4- 85 1E 2580 STA *TEMP
72E6- 60 2590 RTS
2600 ;
2601 ;DHOR trekt een horizontale lijn tussen (X1,Y1) en (X2,Y2).
2602 ;
72E7- 20 36 70 2610 DHOR JSR PLOT ; REPEAT
72EA- 8A 2620 TXA
72EB- F0 1A 2630 BEQ HORB
72ED- E6 10 2640 INC *X1L
72EF- D0 02 2650 BNE HORA ; PLOT(X1,Y1)
72F1- E6 11 2660 INC *X1H
72F3- 38 2670 HORA SEC
72F4- A5 19 2680 LDA *DXL ; X1 := X1 + 1
72F6- E9 01 2690 SBC #1
72F8- 85 19 2700 STA *DXL
72FA- A5 1A 2710 LDA *DXH
72FC- E9 00 2720 SBC #0 ; DX := DX - 1
72FE- 85 1A 2730 STA *DXH

```



```

7300- 05 19      2740      ORA *DXL
7302- D0 E3      2750      BNE DHOR
7304- 4C 36 70    2760      JMP PLOT ; UNTIL X1 veelvoud van 8
7307- A5 19      2770      LDA *DXL
7309- 46 1A      2780      LSR *DXH
730B- 6A         2790      ROR A
730C- 46 1A      2800      LSR *DXH
730E- 6A         2810      ROR A
730F- 4A         2820      LSR A ; DX := DX / 8
7310- 85 19      2830      STA *DXL ; (DX := aantal bytes tussen X1 en X2)
7312- F0 34      2840      BEQ HORD ; WHILE DX <> 0 DO
7314- A9 FF      2850      LDA #$FF
7316- 2C 82 19    2860      BIT $1982 ; wacht op een sync
7319- 30 FB      2870      BMI HORD1
731B- 91 13      2880      STA (AL),Y ; zet 8 puntjes weg
              2889 ;
731D- A5 14      2920      LDA *AH ; IF ADH,ADL = $CFFB...$CFFF
731F- C9 CF      2930      CMP #$CF
7321- D0 16      2940      BNE INCBA
7323- A5 13      2950      LDA *AL
7325- 29 FB      2960      AND #$FB
7327- C9 FB      2970      CMP #$FB
7329- D0 0E      2980      BNE INCBA
              2990 ;
732B- AD 80 19    3000      LDA $1980 ; THEN verwissel van bank
732E- 29 FD      3010      AND #$FD
7330- 09 01      3020      ORA #1
7332- 8D 80 19    3030      STA $1980
7335- A9 AF      3050      LDA *AF
7337- 85 14      3060      STA *AH
              3070 ;
7339- 18         3080      CLC
733A- A5 13      3090      LDA *AL ; schuif 8 plaatsen op
733C- 69 08      3100      ADC #8
733E- 85 13      3110      STA *AL
7340- 90 02      3120      BCC INCBE
7342- E6 14      3130      INC *AH
              3135 ;
7344- C6 19      3140      DEC *DXL ; DX := DX - 1
7346- D0 CC      3150      BNE HORD ; ENDWHILE
              3155 ;
7348- A5 16      3160      LDA *X2L ; maak einde v/d lijn af
734A- 29 07      3170      AND #7
734C- AA         3180      TAX
734D- A9 00      3190      LDA #0
734F- 38         3200      SEC
7350- FD D3 70    3210      SBC PLOTTAB,X
7353- 4C 39 70    3220      JMP PW1
              3221      .CT

```

F04 0E23 41F0-5013

```

4730 ;
4731 ; De tabel SINTAB bevat de sinuscurve van 0-90 graden.
4732 ; M.b.v. deze tabel worden de sinus- en cosinuswaarden berekend.
4740 ;
4750 SINTAB .BY $00 $06 $0C $13 $19 $1F $25 $2C
7356- 00 06 0C
7359- 13 19 1F
735C- 25 2C
735E- 32 38 3E 4760 .BY $32 $38 $3E $44 $4A $50 $56 $5C
7361- 44 4A 50
7364- 56 5C
7366- 62 68 6D 4770 .BY $62 $68 $6D $73 $78 $7E $83 $89
7369- 73 78 7E
736C- 83 89
736E- 8F 93 98 4780 .BY $8F $93 $98 $9D $A2 $A7 $AC $B0
7371- 9D A2 A7
7374- AC B0
7376- B5 B9 BD 4790 .BY $B5 $B9 $BD $C1 $C5 $C9 $CD $D1
7379- C1 C5 C9
737C- CD D1

```

```

737E- D4 DB DB 4800      .BY $D4 $DB $DB $DE $E1 $E4 $E7 $E9
7381- DE E1 E4
7384- E7 E9
7386- EC EE F0 4810      .BY $EC $EE $F0 $F2 $F4 $F6 $F8 $F9
7389- F2 F4 F6
738C- F8 F9
738E- FA FB FC 4820      .BY $FA $FB $FC $FD $FE $FE $FF $FF $FF
7391- FD FE FE
7394- FF FF FF

4830 ;
4840 F1 .DE $20 ; 1ste hoek voor de gonio-functies
4850 F2 .DE $21 ; 2de hoek voor de arc-functie
4860 FAC1H .DI DIVI ; accumulator1 voor MULTIPLY
4870 FAC1L .DI DIVIL ; accumulator2 voor MULTIPLY
4880 FAC2 .DI DIVRL ; accumulator2 voor MULTIPLY
4890 PRSCR .DI AL ; scratch-geheugen
4900 PRL .DI X1L ; produkt van FAC1 en FAC2
4910 PRH .DI X1H

4920 ;
4930 ;SIN(F1)
4940 ;INPUT : F1 is (HOEK/1.41)
4950 ;OUTPUT : ACCU --> 255*SIN(F1) LSB
4960 ;Y-reg --> 255*SIN(F1) SIGN ($00+= $FF=-)
4961 ;
4962 ;Voorbeeld:
4963 ;Stel de hoek v/d gewenste sinus is 14.1 graden,
4964 ;dan SIN F1 aanroepen met F1 = 14.1/1.41 = 10 = $0A.
4965 ;Y-reg, ACCU := 255*SIN(14.1) = $003E = 62
4966 ;
4967 ;Voor 255*SIN(360-14.1) geldt:
4968 ;Y-reg, ACCU := $FF3E = -62
4970 ;
7397- A0 00 4980 SIN F LDY #0 ; SIGN := $00
7399- A5 20 4990 LDA #F1
739B- 10 04 5000 SIN F1 BPL SIN F2 ; IF F1 > 180 graden THEN
739D- 29 7F 5010 AND #7F ; F1 := F1 - 180 graden
739F- A0 FF 5020 LDY #FF ; SIGN := $FF
73A1- C9 41 5030 SIN F2 CMP #41 ; IF F1 > 90 graden THEN
73A3- 30 05 5040 BMI SIN F3 ; BEGIN
73A5- 49 FF 5050 EOR #FF ; draai de
73A7- 18 5060 CLC ; tabel om
73A8- 69 81 5070 ADC #81 ; END
73AA- AA 5080 SIN F3 TAX
73AB- BD 56 73 5090 LDA SINTAB,X ACCU := 255*SIN(F1)
73AE- 60 5100 RTS

5110 ;
5120 ;COS(F1) DMV SIN(F1+90graden)
5130 ;
73AF- A0 00 5140 COS F LDY #0
73B1- A5 20 5150 LDA #F1
73B3- 18 5160 CLC
73B4- 69 40 5170 ADC #40
73B6- 4C 9B 73 5180 JMP SIN F1

5190 ;
5200 ;Multiply:
5201 ;
5202 ;INPUT : FAC1H,FAC1L
5203 ; FAC2
5204 ; Y-reg (is SIGN, positief indien Y=0)
5205 ;OUTPUT: PRH,PRL (two's complement)
5206 ;
5210 ;PRH,PRL := SIGN * (FAC1H,FAC1L * FAC2) / 256
5220 ;
73B9- A9 00 5230 MUL LDA #0
73BB- 85 13 5240 STA #PRSCR ; PRSCR := 0
73BD- 85 10 5250 STA #PRL ; PR := 0
73BF- 85 11 5260 STA #PRH
73C1- 85 1B 5270 STA #FAC1H ; FAC1H := 0
73C3- A2 08 5280 LDX #8 ; X := 8
73C5- 46 19 5290 MUL1 LSR #FAC2 ; REPEAT
73C7- 90 0D 5300 BCC MUL2 ; FAC2 := FAC2 / 2
73C9- 1B 5310 CLC ; IF C=0 THEN

```

```

73CA- A5 1F 5320 LDA #FAC1L ; BEGIN
73CC- 65 13 5330 ADC #PRSCR ; PRL,PRSCR:=PRL,PRSCR+FAC1
73CE- 85 13 5340 STA #PRSCR
73D0- A5 1B 5350 LDA #FAC1H
73D2- 65 10 5360 ADC #PRL
73D4- 85 10 5370 STA #PRL ; END
73D6- 06 1F 5380 MUL2 ASL #FAC1L ; FAC1 := FAC1 * 2
73D8- 26 1B 5390 ROL #FAC1H
73DA- CA 5400 DEX ; X := X - 1
73DB- D0 EB 5410 BNE MUL1 ; UNTIL X=0
5420 ;
5450 ;Maak two's complement
5452 ;
73DD- C0 00 5460 CPY #00
73DF- F0 1A 5470 BEQ MULE
73E1- A5 13 5480 LDA #PRSCR
73E3- 49 FF 5490 EOR #$FF
73E5- 85 13 5500 STA #PRSCR
73E7- A5 10 5510 LDA #PRL
73E9- 49 FF 5520 EOR #$FF
73EB- 85 10 5530 STA #PRL
73ED- A9 FF 5540 LDA #$FF
73EF- 85 11 5550 STA #PRH
73F1- E6 13 5560 INC #PRSCR
73F3- D0 06 5570 BNE MULE
73F5- E6 10 5580 INC #PRL
73F7- D0 02 5590 BNE MULE
73F9- E6 11 5600 INC #PRH
73FB- 60 5610 MULE RTS
5620 ;
5630 ;
5640 ;CIRCLE:
5641 ;
5642 ;Plot een cirkel op het scherm met straal R en
5643 ;middelpunt X2,Y2.
5644 ;
*****
5650 ;INPUT: X2L,X2H Y2 R
5660 ;
73FC- A9 00 5670 CIRCLE LDA #0
73FE- 85 20 5680 STA #F1
7400- 85 21 5690 STA #F2
5700 ;
5710 ;ARC:
5711 ;
5712 ;Plot een boog op het scherm met straal R en
5713 ;middelpunt X2,Y2 van hoek F1 tot hoek F2.
5714 ;
5720 ;INPUT: X2L,X2H Y2 R F1 F2
5721 ;
5722 ;Gebruikte procedure:
5723 ;
5724 ;REPEAT
5725 ; X1H,X1L := X2H,X2L + R*COS(F1)
5726 ; Y1 := Y2 - R*K*SIN(F1)
5727 ; { K is een resolutie-korrektiefactor }
5728 ; { K=177/256 }
5729 ; PLOT(X1,Y1)
5730 ; F1 := F1 + 1
5731 ; UNTIL F1=F2
5732 ;
7402- A5 1D 5740 ARC LDA #R ; REPEAT
7404- 85 19 5750 STA #FAC2
7406- 20 97 73 5760 JSR SINF ; A := 255*SIN(F1)
7409- 85 1F 5770 STA #FAC1L
740B- 46 1F 5780 LSR #FAC1L
740D- 46 1F 5790 LSR #FAC1L ; FAC1 := 63*SIN(F1)
740F- 38 5800 SEC
7410- E5 1F 5810 SBC #FAC1L ; A := 192*SIN(F1)
7412- 46 1F 5820 LSR #FAC1L
7414- 46 1F 5830 LSR #FAC1L ; FAC1 := 15*SIN(F1)
7416- 38 5840 SEC
7417- E5 1F 5850 SBC #FAC1L

```



```

7419- 85 1F      5860      STA *FAC1L ; FAC1 := 177*SIN(F1)
741B- 20 B9 73   5870      JSR MUL ; PR := FAC1 * FAC2 / 256
741E- A5 10      5880      LDA *PRL
7420- 85 12      5890      STA *Y1 ; Y1 := R*(177/256)*SIN(F1)
                    5900 ;
7422- A5 1D      5910      LDA *R
7424- 85 19      5920      STA *FAC2 ; FAC2 := R
7426- 20 AF 73   5930      JSR COSF
7429- 85 1F      5940      STA *FAC1L ; FAC1 := 255*COS(F1)
742B- 20 B9 73   5950      JSR MUL ; PR := FAC1 * FAC2 / 256
742E- 18         5970      CLC
742F- A5 10      5980      LDA *PRL ; X1 := X2 + R*COS(F1)
7431- 65 16      5990      ADC *X2L
7433- 85 10      6000      STA *X1L
7435- A5 11      6010      LDA *PRH
7437- 65 17      6020      ADC *X2H
7439- 85 11      6030      STA *X1H
743B- 38         6040      SEC
743C- A5 18      6050      LDA *Y2 ; Y1 := Y2 - R*(177/256)*SIN(F1)
743E- E5 12      6060      SBC *Y1
7440- 85 12      6070      STA *Y1
                    6080 ;
7442- 20 36 70   6090      JSR PLOT ; PLOT(X1,Y1)
7445- E6 20      6100      INC *F1 ; F1 := F1 + 1
7447- A5 20      6110      LDA *F1
7449- C5 21      6120      CMP *F2
744B- D0 B5      6130      BNE ARC ; UNTIL F1=F2
744D- 60         6140      RTS
                    6150      .EN

```

```

*****
*
*          PROGRAMMA "COMPETITIESTANDEN" HANDBALCOMPETITIE
*          een Basic-programma
*          voor de Elektuur JUNIOR computer
*          met 2 cassetterecorders en een
*          teletype printer
*
*****

```

Door: Gerard Keet

Het programma "Competitiestanden" heeft, gezien de handbalcompetitie die bestaat uit een veld- en zaalgedeelte, de mogelijkheid om voor een van beide te kiezen, en tegen het einde te bekijken of, en zo ja, welke teams kampioenskansen hebben.

Het benodigde geheugen is 32 K. Memory size is 20.000. Basic start op \$2000.

Voor zaal en veld gelden resp. 12 en 8 teams per klasse. Een klassebenaming moet uit 5 posities bestaan.

De A op regel 20010 = max. 36 teams.

De B = aantal gereserveerde plaatsen voor teams (12).

De C = beschikbaar zijn 12 posities voor teamnaam.

De D = gereserveerde ruimte voor klasse aanduiding.

De E = gereserveerd voor wedstrijden, d.w.z.

Gesp.	Ptn.	Voor	Tegen
2 pos	2 pos	3 pos	3 pos

De F op regel 20020 = benodigde ruimte per klasse.

W en P zijn resp. welpen en pupillen.

Jeugd wordt aangeduid met JE, dit daar bij onze competitie geldt dat, indien in deze categorie gelijk geëindigd, beide kampioen!

AT beduidt: aantal teams

Het programma zal worden vervolgd met een Printroutine voor teletype 110 baud en het programma ODIN SUPERTEAM. De laatste is een aardige routine die laat zien welk team het beste gepresteerd heeft. Een en ander in een volgende editie.

Alle programma's zijn ontwikkeld om de handbalvereniging ODIN in Heemskerk van dienst te zijn. Zij draaien reeds twee seizoenen zonder problemen op een JUNIOR-computer met Synertek-Basic met een terminal van ADD's. De Basic lijkt wel wat op de KB-9 Basic.

## DOCUMENTATIE KOMPETITIESTANDEN

- Inhoud: 1 Programmastructuur  
2 Variabelen en konstanten  
3 Aanpassen programma aan specifieke situatie  
4 Gebruikershandleiding

### 1. Programmastructuur

Het hoofdprogrammablok is geprogrammeerd in de regelnummers 20000-25200.

Vanuit dit programmablok wordt naar subroutines gesprongen op basis van de keuze van een bepaalde functie.

<u>Keuze</u>	<u>Functie</u>	<u>Regelnummers</u>
1	Standen inlezen	18000-18090
2	Klasse toevoegen	17000-17290
3	Klasse wijzigen	16000-16240
4	Klasse verwijderen	15000-15090
5	Uitslagen invoeren	14000-14650
6	Standen printen	13000-13280
7	Standen wegschrijven	19000-19090
8	Initialiseren standen	5000- 5090

Functie 3 "Klasse wijzigen" presenteert als enige functie weer een keuzescherf, op basis waarvan weer naar subroutines gesprongen wordt.

<u>Keuze</u>	<u>Wijzigfunctie</u>	<u>Regelnummers</u>
1	Klassenummer	3100-3190
2	Team toevoegen	8000-8090
3	Team verwijderen	7000-7090
4	Team wijzigen	6000-6150

Vanuit functie 6 "Standen printen" kan, als de stand op het scherm staat, desgewenst kampioensinformatie opgevraagd worden. Het programmadeel dat dit verzorgt is ook een subroutine en geprogrammeerd in de regelnummers 30000-30450.

De overige subroutines zijn routines die vanuit verschillende plaatsen aangeroepen worden.

<u>Subroutine</u>	<u>Regelnummers</u>
Invoeren teamnaam	3000-3090
Invoeren klassenaam	3100-3190

## 2. Variabelen en konstanten

Om niet te snel in geheugenruimteproblemen te komen, zijn korte namen gekozen. Dit maakt het programma minder goed leesbaar. Hierbij een verklaring van de gebruikte namen.

<u>Naam</u>	<u>Waarde</u>	<u>Omschrijving</u>
A	36	Aantal klassen waarvoor ruimte in de buffer gereserveerd is.
B	12	Aantal teams per klasse (maximum).
C	12	Lengte van de teamnaam (maximum).
D	5	Lengte van de klassenaam (maximum).
E	10	Aantal posities nodig voor de stand van 1 team, nl. 2 voor "gespeeld", 2 voor "punten", 3 voor "doelpunten voor" en 3 voor "doelpunten tegen".
F	$D+B*(C+E)$	Benodigde ruimte per klasse in de buffer.
AI	24576	Beginadres buffer (\$6000)
LA	34259	Laatste adres buffer
S\$(x)	stringvar	Tabel voor de teamnamen van de te verwerken klasse.
S(x,y)	var	Tabel voor de standen van de te verwerken klasse.
G\$,K\$,P\$,T\$	stringvar	Lokale hulpvariabelen.
U\$,V\$,Z\$	stringvar	Lokale hulpvariabelen.
H,I,J,K,L	var	Lokale hulpvariabelen.
R,T,U,X	var	Lokale hulpvariabelen.
AT	var	Aantal teams van de te verwerken klasse.
SKL	var	Lokale variabele (LV) voor het vasthouden van de oorspronkelijke waarde van KL (SaveKL).
KL	var	Beginadres van de te verwerken klasse in de buffer.
H\$	stringvar	Hulpveld voor lezen/schrijven van/naar buffer.
AA	var	Aantal te spelen wedstrijden per team.
NWKL	boolean var	Globale hulpvariabele (GV) om in subroutine te herkennen of het om een nieuwe klasse gaat.
NG	boolean var	GV om vanuit een subroutine de waarde "niet gevonden" over te dragen.
JE	boolean var	LV voor herkenning van "jeugd".
KA	boolean var	LV voor vastleggen of een team kampioen is.
NK	boolean var	LV voor vastleggen of een team niet kampioen is.
GB	var	LV voor bijhouden "gespeeld beste".
PB	var	LV voor bijhouden "punten beste".
TS\$	stringvar	LV voor vastleggen eerstvolgende tegenstander.
OD	var	LV voor vastleggen plaats ODIN in tabel.
GT	var	LV voor bijhouden "gespeeld tegenstander".
PT	var	LV voor bijhouden "punten tegenstander".

## 3. Aanpassen programma aan specifieke situatie

Er is een goede kans dat er voor sommige 6502-ers ruimtegebrek ontstaat tijdens het werken met het programma. Bewaar in dat geval als documentatie een listing die alle remarks bevat, maar verwijder ze uit het programma waar je mee werkt.

Door de modulaire opbouw van het programma is het ook heel goed mogelijk functies (subroutines) apart te programmeren, bv. een printprogramma. Je kan dan met het hoofdprogramma de standen inlezen, waarna je het printprogramma (incl. kampioensinfo) laadt en de standen kan printen.

Het programma is speciaal geschreven voor handbalvereniging ODIN uit Heemskerk.

Specifieke kenmerken voor deze vereniging/kompetitie zijn:

- Er worden minimaal twaalf wedstrijden per competitie gespeeld, zonodig door anderhalve of dubbele competitie.
- Bij jeugd (pupillen=P en welpen=W) zijn bij gelijk eindigen op de eerste plaats van meerdere teams deze alle kampioen.

E.e.a. is voor de standen niet van belang, maar wel voor de kampioensinformatie.

1. Het aantal wedstrijden per competitie per team wordt geregeld in regelnummer 10300. Zonodig moeten deze regel dus aangepast worden.

2. Vanzelfsprekend moet overal "ODIN" vervangen worden door de naam van de vereniging waarvoor de standen bijgehouden worden.

3. Op regel 30010 wordt bepaald of er sprake is van "jeugd", t.b.v. de genoemde uitzonderingssituatie. Hiertoe wordt de tweede positie van de klassenaam afgevraagd, die in ons geval de categorie aanduiding bevat. Als er in jouw geval geen sprake is van een dergelijke uitzondering dan kan op regel 30010 het tweede statement weggelaten worden, en verder de regels 30160, 30170, 30300-30330 en 30370-30400.

Als je de uitzondering wilt laten zitten voor eventueel toekomstig gebruik, dan kan je hem uitschakelen door het tweede statement op regel 30010 te wijzigen in "JE=FALSE". In verwerkingstijd maakt dit nauwelijks uit, wel in geheugenruimtebeslag.

4. De benodigde geheugenruimte voor de buffer is bijna 10K (9684). In ons geval begint de buffer op adres 24576 (\$6000). We hebben nl. 36K beschikbaar. We starten Basic op met memory-size=20000. Nogal slordig dus, maar wij kennen in dit geval geen ruimteproblemen. Als je daar wel last van hebt, dan kan je e.e.a. wat nauwkeuriger doen.

Vb: Als je 24K geheugenruimte hebt (24576) dan kan je de beginwijzer van de buffer op  $24575 - 9684 + 1 = 14892$  zetten. Dus  $AI=14892$  en je kan dan Basic opstarten met memory-size=14891. Vergeet niet in de routine "Standen wegschrijven" de adressen aan te passen, in dit geval "POKE 6768,44:POKE 6769,148:POKE 6770,255:POKE 6771,95", maak  $LA=24575$ .

## 4. Gebruikershandleiding

### 4.1 Opstarten Basic

Start op de gebruikelijke manier Basic. Voer op de vraag "memory-size" "20000" in. Vraagt het systeem hier niet om, geef dan na "OK" het commando "clear,20000".

### 4.2 Laden programma

Laad het programma op de gebruikelijke wijze en start (met RUN).



## 4.3 Werken met het programma

Na het run-commando verschijnt het keuzeschermb. Deze beschrijving volgt dit keuzeschermb. Derhalve wordt achtereenvolgens beschreven:

1. Standen inlezen
2. Klasse toevoegen
3. Klasse wijzigen
4. Klasse verwijderen
5. Uitslagen invoeren
6. Standen printen
7. Standen wegschrijven
8. Initialiseren standen
9. Stoppen

Als voor de eerste keer met het programma gewerkt wordt, moet eerst de buffer geïnitieerd worden met functie 8, daarna met functie 2 de klassen inbrengen. Pas daarna kunnen uitslagen ingevoerd worden, enz.

### 4.3.1 Standen inlezen

Zorg dat de "leesrecorder" klaar staat met het goede bandje en op de goede plaats. Na keuze 1 worden dan de standen in de komputer ingelezen. Als dit gebeurd is verschijnt het keuzeschermb weer.

### 4.3.2 Klasse toevoegen

Deze functie moet gebruikt worden als voor de aanvang van een nieuwe competitie, de nieuwe klassen met hun teams ingevoerd moeten worden. Zorg wel dat er eerst geïnitieerd is (keuze 8).

Na keuze 2 gedaan te hebben, kan de komputer met de melding "Geen ruimte voor nieuwe klasse" komen. Dit betekent dat alle 36 plaatsen al gebruikt zijn, of dat er niet geïnitieerd is. Het keuzemenu verschijnt weer.

Is alles in orde dan vraagt de komputer om een klassennummer. Als dit nummer al bestaat, dan wordt dat gemeld en verschijnt het keuzemenu. Bestaat de klasse nog niet dan moet het aantal teams van die klasse ingevoerd worden, gevolgd door de namen van de teams. Als dit gebeurd is, kan een volgende klasse toegevoegd worden. Zo niet dan verschijnt het keuzemenu weer.

### 4.3.3 Klasse wijzigen

Deze functie hoeft, als alles altijd regulier verloopt, niet gebruikt te worden (systeembewakingsfunctie). Er zijn echter situaties waarin iets aangepast en/of gerepareerd moet worden, b.v. als je achteraf tot de ontdekking komt dat je foute uitslagen ingevoerd hebt. Soms moet halverwege de competitie een team verwijderd worden, omdat dat teruggetrokken wordt, of juist toegevoegd worden. Voor aanpassingen moet keuze 3 ingetoetst worden. Na invoeren van de klassenaam (-nummer) verschijnt het "keuzeschermb wijzigen", waar gekozen kan worden uit:

1. Klassennummer
2. Team toevoegen
3. Team verwijderen
4. Team wijzigen
5. Stop

Na een verwerkte wijziging wordt gevraagd of een andere klasse gewijzigd moet worden. Na "J" moet een nieuwe klassenaam opgegeven worden, na "N" niet. In beide gevallen verschijnt "keuzescherf wijzigen" weer.  
Pas na "5 Stop" verschijnt het (hoofd-)keuzemenu weer.

#### 4.3.3.1 Klassenummer

Met keuze 1 kan de klassenaam gewijzigd worden.

#### 4.3.3.2 Team toevoegen

Met keuze 2 kan een team aan de klasse toegevoegd worden. Als er al twaalf teams in de klasse zitten, komt de melding "Geen ruimte voor nieuw team".

#### 4.3.3.3 Team verwijderen

Met keuze 3 kan een team uit de klasse verwijderd worden. Als het team niet in de klasse voorkomt, wordt dit gemeld.

#### 4.3.3.4 Team wijzigen

Hiermee kunnen de resultaten van een team gewijzigd worden. Na invoeren van de teamnaam moet resp. gespeelde wedstrijden, behaalde punten, doelpunten gescoord en tegendoelpunten ingevoerd worden. Als het team niet in de klasse voorkomt wordt dit gemeld.

#### 4.3.4 Klasse verwijderen

Deze functie dient om een hele klasse uit de standen te verwijderen. Dit kan van pas komen als een team van je eigen vereniging uit de competitie teruggetrokken wordt, waardoor de klasse voor jou niet meer van belang is. Omdat dit een nogal rigoureuze activiteit is, wordt na invoeren van de klassenaam eerst nog eens gevraagd of je wel zeker weet dat die klasse weg moet. Als je dan "J" invult gebeurt het ook onherroepelijk.

#### 4.3.5 Uitslagen invoeren

Met deze functie moeten steeds de uitslagen van de gespeelde wedstrijden ingevoerd worden. Na de klasse ingevoerd te hebben, moeten achtereenvolgens het thuisspelende team, het bezoekende team, het aantal doelpunten voor en het aantal doelpunten tegen ingevoerd worden. Als bij het invoeren "thuis" of "uit" een fout gemaakt wordt voordat <return> ingetoetst is, kan door invoeren van een "\*" opnieuw begonnen worden met invoeren.

B.v.: Je wilt invoeren "Thuis: <FC Knudde 13>" en je voert in "<SC Knudde..." en constateert dan dat het niet goed is. Voer dan een "\*" in op de plaats waar je gebleven bent, gevolgd door <return>:"<SC Knudde\*><return>". De computer vraagt dan om opnieuw "Thuis" in te voeren. Deze mogelijkheid voorkomt dat je in dit soort gevallen een heel stuk moet "backspacen".

De computer voert een aantal controles op de ingevoerde gegevens uit en meldt het als er iets mis is. Als alles goed bevonden wordt, wordt de uitslag geaccepteerd en kan de volgende uitslag van deze klasse ingevoerd worden. Als van alle klassen de uitslagen zijn ingevoerd moet bij "klasse" een "\*" ingevoerd worden. Het keuzemenu verschijnt dan weer.

## 4.3.6 Standen printen

Met keuze 6 kan je de standen op het beeldscherm bekijken. Na de klasse opgegeven te hebben, zoekt de computer deze op, gaat de stand formen en presenteert deze op het scherm. Je krijgt dan de vraag of je wel of geen kampioensinformatie wenst. Als je dat niet wilt geef je "N" op en komt de vraag of je nog een klasse wilt zien. Bij "N" verschijnt het keuzeschermbij "J" wordt weer om een klasse gevraagd, enz.

Als je wel kampioensinformatie wilt (dit is pas zinvol tegen het eind van de competitie) dan geef je "J" op en wordt om de volgende tegenstander van je verenigingsteam gevraagd. De computer meldt dan hoe de kampioensansen van dit team liggen, waarna gevraagd wordt of je nog een klasse wilt zien, enz.

## 4.3.7 Standen wegschrijven

Met functie 7 kan je de bijgewerkte standen weer naar de band schrijven. Zorg dat de opnamerecorder klaar staat met de goede band op de goede plaats.

Denk er ook aan een "back-up" te maken, het zou zonde zijn als later door een dwarse bit een hoop invoerwerk overgedaan moet worden.

Als de standen weggeschreven zijn, verschijnt het keuzeschermbij "J" wordt weer om een klasse gevraagd, enz. Je kan dan gewoon verder gaan met printen etc. want de standen staan nog steeds in de buffer. Het is dan ook aan te bevelen direkt na het invoeren van de uitslagen de standen weg te schrijven. Mocht er dan iets mis gaan dan staan die alvast veilig op de band.

## 4.3.8 Initialiseren standen

Met deze functie zorg je dat de geheugenbuffer "leeg" gemaakt wordt. Dit is nodig als je voor aanvang van een nieuwe competitie weer nieuwe klassen met teams moet invoeren. Verder nooit gebruiken, want voor je het weet ben je je standen kwijt. Dat is natuurlijk niet erg als de bijgewerkte standen al op de band staan, alhoewel het toch hinderlijk is als je ze weer opnieuw moet inlezen.

Om vergissingen uit te sluiten vraagt de komputer eerst nog of je zeker weet dat er geïnitialiseerd moet worden.

## 4.3.9 Stoppen

Als je er genoeg van hebt kies je "9", je zit dan weer in "BASIC".

Kom je tot de ontdekking dat je toch nog verder wilt werken, dan geef je "Run".

Je hoeft niet eerst de standen in te lezen want de buffer is nog steeds ongewijzigd.

78 88 MLIST

SCR # 78

```

0 ( FORTH SCREEN EDITOR )
1 (      Fridus Jonkman , Stijn Streuvelsstraat 9 )
2 (      5242 GD Rosmalen , tel. 04192-16146 )
3 ( ===== )
4 ( De line-editor van W. Ragsdale voor fig-FORTH is niet erg )
5 ( plezierig in het gebruik: reden om een full-screen-editor )
6 ( te gaan bouwen. Ik heb gebruik gemaakt van mijn publikatie )
7 ( met woorden voor cursorbesturing en simpele grafische routi- )
8 ( nes voor de Junior met VDU-kaart. In deze vorm is de editor )
9 ( geschikt voor fig-systemen met diskbuffers kleiner dan 1K. )
10 ( Omdat de diskbuffers elk een tussenruimte hebben van 4 bytes, )
11 ( heb ik een extra buffer gemaakt van 1K, waarin tekst wordt )
12 ( gemanipuleerd. Via de constante TEMP krijgt deze een plaats. )
13 ( In de laatste screen wordt aangegeven hoe je de editor een- )
14 ( voudig geschikt kunt maken voor systemen met diskbuffers van )
15 ( 1K. )

```

SCR # 79

```

0 FORTH DEFINITIONS HEX
1 ( \ (( --- )) : commentaar na \ i.o.v. tussen haakjes )
2 : \ IN @ C/L / 1+ C/L * IN ! : IMMEDIATE
3 \ WHERE en COPY komen uit de Ragsdale-editor
4 \ WHERE (( block# offset --- ))
5 \ gebruiken na error bij comoilieren van screens
6 : WHERE DUP B/SCR / DUP SCR ! " SCR # " DECIMAL . SWAP C/L /MOD
7 C/L * ROT BLOCK + CR C/L TYPE CR HERE C@ - SPACES SE
8 EMIT [COMPILE] QUIT ;
9 \ COPY (( van-scr# naar-scr# --- ))
10 : COPY B/SCR * OFFSET @ + SWAP B/SCR * B/SCR OVER + SWAP DO DUP
11 FORTH I BLOCK 2 - ! 1+ UPDATE LOOP DROP FLUSH ;
12 \ CMOVE (( van naar #bytes --- ))
13 \ kan i.t.t. CMOVE ook blokken overlappend vooruit schuiven
14 : CMOVE -DUP IF >R R + 1 - SWAP 1 - R) OVER + DO I C@ OVER C!
15 1 - -1 +LOOP DROP ELSE DROP DROP ENDIF ; --)

```

SCR # 80

```

0 \ Gebruik is gemaakt van de CASE-konstruktie van Charles Eaker
1 : CASE ?COMP CSP @ !CSP 4 : IMMEDIATE
2 : OF 4 ?PAIRS COMPILE OVER COMPILE = COMPILE OBRANCH
3 HERE 0 , COMPILE DROP 5 : IMMEDIATE
4 : ENDOF 5 ?PAIRS COMPILE BRANCH HERE 0 , SWAP 2
5 [COMPILE] ENDIF 4 : IMMEDIATE
6 : ENDCASE 4 ?PAIRS COMPILE DROP BEGIN SP@ CSP @ = 0= WHILE 2
7 [COMPILE] ENDIF REPEAT CSP ! : IMMEDIATE
8 \ Specifieke editorwoorden komen in EDIT
9 VOCABULARY EDIT IMMEDIATE EDIT DEFINITIONS DECIMAL
10 512 CONSTANT TEMP \ de extra buffer begint hier op $0200
11 0 VARIABLE CUR \ cursorpositie t.o.v. TEMP
12 0 VARIABLE INSERT \ insert-mode = 1
13 \ DISPL-I-ON (( --- )) : display insert-on
14 : DISPL-I-ON 63 1 !CURSOR " INSERT-ON"
15 10 1 62 0 RECTANGLE : --)

```

SCR # 81

```

0 \ DISPL-I-OFF (( --- )) : display soaties
1 : DISPL-I-OFF 3 0 DO 62 I !CURSOR 14 SPACES LOOP :
2 \ CUR)CHAR-LINE (( --- char# line# ))
3 \ zet de cursorpositie om naar x-y-waarden voor de screen
4 : CUR)CHAR-LINE CUR @ C/L /MOD :
5 \ .CUR (( --- )) : display cursor
6 : .CUR CUR)CHAR-LINE 5 + >R 9 + R) !CURSOR :
7 \ DISPL-LINE (( line# --- )) : display line
8 : DISPL-LINE 9 OVER 5 + !CURSOR
9 C/L * TEMP + C/L TYPE ;
10 \ DISPL-SCR (( line# --- )) : display screen vanaf line#
11 : DISPL-SCR 16 SWAP DO I DISPL-LINE LOOP :
12 \ GET-SCR (( --- )) : haal screen naar diskbuffer
13 \ en naar extra buffer
14 : GET-SCR SCR @ 16 0 DO I OVER (LINE) TEMP I C/L * + SWAP
15 CMOVE LOOP DROP : --)

```



```

SCR # 82
0 \ SETUP (( --- )) : display scherm-lay-out editor
1 : SETUP CLS ( clear screen ) 10 1 !CURSOR
2 " F o r t h   S c r e e n   E d i t o r "
3 37 1 8 0 RECTANGLE
4 9 3 !CURSOR " SCR# " SCR ? \ display scr-nr
5 16 0 DO 5 1 OVER + !CURSOR 1 2 .R LOOP \ display
6 \ line-nr's
7 64 16 8 4 RECTANGLE 0 INSERT !
8 0 22 !CURSOR ; \ vanaf deze positie kan je informatie
9 \ kwijs over te gebruiken kommando's
10 \ nu volgen alle commando's die kunnen worden gegeven
11 \ MODE (( --- )) : toggle mode en display
12 : MODE INSERT @ 0= DUP INSERT ! \ toggle
13 IF DISPL-I-ON ELSE DISPL-I-OFF ENDIF \ display
14 .CUR ;
15 -->

```

```

SCR # 83
0 \ HOME (( --- )) : cursor-home binnen screen
1 : HOME 9 5 !CURSOR 0 CUR ! ;
2 \ CLEAR (( --- )) : clear/display screen en cursor-home
3 : CLEAR TEMP 1024 BLANKS 0 DISPL-SCR HOME ;
4 \ LEFT (( --- )) : cursor-left binnen screen
5 : LEFT CUR>CHAR-LINE DROP 0 ) IF -1 CUR +! 8 EMIT ENDIF ;
6 \ RIGHT (( --- )) : cursor-right binnen screen
7 : RIGHT CUR>CHAR-LINE DROP 63 ( IF 1 CUR +! 9 EMIT ENDIF ;
8 \ UP (( --- )) : cursor-up binnen screen
9 : UP CUR>CHAR-LINE SWAP DROP 0 ) IF -64 CUR +! 11 EMIT ENDIF ;
10 \ DOWN (( --- )) : cursor-down binnen screen
11 : DOWN CUR>CHAR-LINE SWAP DROP 15 ( IF 64 CUR +! 10 EMIT ENDIF ;
12 \ I-LINE (( --- )) : voeg lege regel tussen op cursorpositie
13 : I-LINE CUR>CHAR-LINE SWAP DROP DUP C/L * TEMP + DUP DUP C/L
14 + OVER 1024 TEMP + SWAP - (CMOVE C/L BLANKS
15 DISPL-SCR .CUR ; -->

```

```

SCR # 84
0 \ D-LINE (( --- )) : verwijder regel op cursorpositie
1 : D-LINE CUR>CHAR-LINE SWAP DROP DUP C/L * TEMP + DUP C/L +
2 SWAP OVER 1024 TEMP + SWAP - CMOVE 960 TEMP + C/L
3 BLANKS DISPL-SCR .CUR ;
4 \ H-LINE (( --- )) : bewaar regel op cursorpositie in PAD
5 : H-LINE CUR>CHAR-LINE SWAP DROP C/L * TEMP + PAD 1+ C/L DUP
6 PAD C! CMOVE ;
7 \ R-LINE (( --- )) : PAD naar regel op cursorpositie
8 : R-LINE CUR>CHAR-LINE SWAP DROP DUP PAD 1+ SWAP C/L * TEMP
9 + C/L CMOVE DISPL-LINE .CUR ;
10 \ (ERASE (( --- )) : spaties in regel voor cursorpositie
11 : (ERASE CUR>CHAR-LINE SWAP DROP DUP C/L * TEMP + DUP CUR @
12 TEMP + SWAP - DUP IF BLANKS DISPL-LINE .CUR
13 ELSE DROP DROP DROP ENDIF ;
14 -->
15

```

```

SCR # 85
0 \ ERASE (( --- )) : spaties in regel vanaf cursorpositie
1 : ERASE CUR>CHAR-LINE SWAP DROP DUP CUR @ TEMP + DUP ROT 1+
2 C/L * TEMP + SWAP - BLANKS DISPL-LINE .CUR ;
3 \ D-CHAR (( --- )) : verwijder karakter op cursorpositie
4 : D-CHAR CUR @ TEMP + DUP 1+ SWAP CUR>CHAR-LINE SWAP DROP DUP
5 )R 1+ C/L * TEMP + 1 - DUP )R OVER - CMOVE BL R) C! R)
6 DISPL-LINE .CUR ;
7 \ NEXT-LINE (( --- )) : cursor naar begin volgende regel
8 : NEXT-LINE CUR>CHAR-LINE SWAP DROP DUP 15 ( IF 1+ C/L * CUR !
9 .CUR ELSE DROP ENDIF ;
10 \ CHAR (( ascii-waarde --- )) : voeg karakter toe op cursor-
11 positie en schuif rest van regel op indien insert = on
12 : CHAR CUR @ TEMP + INSERT @ IF DUP DUP 1+ CUR>CHAR-LINE SWAP
13 DROP 1+ C/L * TEMP + OVER - (CMOVE C! ELSE C! ENDIF
14 CUR>CHAR-LINE )R 63 ( IF 1 CUR +! ENDIF R)
15 DISPL-LINE .CUR ;

```

```

SCR # 86
0 EDIT DEFINITIONS DECIMAL \ i.v.m. laden tweede deel
1 \ SAVE (( --- )) : schrijf screen naar (pseudo-)disk(-ram)
2 : SAVE SCR @ B/SCR * DUP B/SCR + SWAP DO TEMP I B/SCR MOD B/BUF
3 * + I BUFFER B/BUF CMOVE UPDATE LOOP FLUSH :
4 \ COMMAND (( --- )) : dit is de hoofdloop van de editor
5 : COMMAND BEGIN KEY DUP 32 < OVER 127 > OR 0= IF DUP CHAR ENDIF
6 DUP 21 - WHILE CASE
7 2 OF H-LINE ENDOF 3 OF R-LINE ENDOF 4 OF I-LINE ENDOF
8 5 OF D-LINE ENDOF 6 OF D-CHAR ENDOF 7 OF (ERASE ENDOF
9 8 OF LEFT ENDOF 9 OF RIGHT ENDOF 10 OF DOWN ENDOF
10 11 OF UP ENDOF 12 OF CLEAR ENDOF 13 OF NEXT-LINE ENDOF
11 16 OF SAVE ENDOF 17 OF ERASE ENDOF 22 OF MODE ENDOF
12 28 OF HOME ENDOF
13 ENDCASE REPEAT DROP :
14
15

```

```

SCR # 87
0 FORTH DEFINITIONS
1 \ Met een van beide volgende commando's kan je vanuit FORTH de
2 \ editor starten.
3 \ E (( scr# --- )) of (( --- )) : indien de stack leed is.
4 \ wordt de te editen screen uit SCR gehaald
5 : E EDIT DEPTH IF SCR ! ENDF SETUP GET-SCR
6 0 DISPL-SCR HOME COMMAND CLS FORTH :
7 \ EC (( scr# --- )) of (( --- )) : als E. maar de screen
8 \ wordt eerst schoon gemaakt
9 : EC EDIT DEPTH IF SCR ! ENDF SETUP CLEAR
10 COMMAND CLS FORTH :
11 ;S
12
13
14
15

```

```

SCR # 88
0 \ Aanpassingen voor FORTH-svsystemen met diskbuffers van 1Kbyte.
1 \
2 \ Nu is geen extra buffer nodig: je kan rechtstreeks editen in
3 \ je diskbuffers !
4 \ 1. De variabele TEMP kan verdwijnen
5 \ 2. Vervang overal TEMP door PREV @
6 \ 3. De definitie van SAVE wordt:
7 \ : SAVE UPDATE FLUSH :
8 \ 4. De definitie van GET-SCR wordt:
9 \ : GET-SCR SCR @ 16 0 DO I OVER (LINE) DROP DROP LOOP DROP :
10
11
12 \
13
14
15

```

OK



Leif Rasmussen, Parkvej 1  
4034 Hørve DK-Danmark

## WORD PROCESSOR PRINTER VERSION

With these changes and extensions of the 'Samson's Wordprocessor v. 2.2 it will be quite easy to write the printers control codes directly into the file.

The controlkeys are optional. You can make your own modifications to suit your keyboard.

Make the changes with the 'monitor', and save them with :

and A\*SA 29,1=0A00/B  
A\*SA 30,1=1200/B

### CONTROL KEYS

^H (arrowkey)..cursor left  
^I (arrowkey)..cursor right  
^J (arrowkey)..cursor down  
^K (arrowkey)..cursor up

^L (homekey)..page 1  
^S .....next page

^DEL (\$1F)..delete line  
^I (\$1C) ...insert line

^W ...write insert mode  
the cursor blink slowly  
this mode enable printing  
chr. \$00 to chr. \$7E  
( - \$08...\$0D )

^Q ...upper case mode  
the cursor blink small  
this mode enable printing  
chr. \$80 to chr. \$FF

ESC ...escape mode  
the cursor blink quickly

ADD 0B9F:	20 1D F7	JSR \$F71D	GET A CHR.
0BA2:	C9 11	CMP #11	IS IT CTRLQ?
0BA4:	D0 15	BNE \$0BBB	IF NOT: LEFT
0BA6:	A2 0A	LDX #0A	TURN
0BAB:	BE 40 E1	STX #E140	ON SMALL
0BAB:	A2 47	LDX #47	BLINKING
0BAD:	BE 41 E1	STX #E141	CURSOR
0BB0:	20 1D F7	JSR \$F71D	GET A CHR.
0BB3:	20 BB 13	JSR #13BB	BLINK OFF
0BB6:	69 7F	ADC #7F	ADD 127
0BB8:	4C F6 0D	JMP \$0DF6	PROC. A CHR.
0BBB:	60	RTS	TO \$0C09
0C06:	20 9F 08	JSR \$0B9F	TO ADD
0C0A:	08	CTRL H ?	GO LEFT
0C14:	09	CTRL I ?	GO RIGHT
0C1E:	0A	CTRL J ?	GO DOWN
0C28:	08	CTRL K ?	GO UP
0C3C:	1C	CTRL I ?	INSERT LINE
0CA7:	1F	CTRL DEL ?	DELETE LINE
0D4C:	17	CTRL W ?	WR. IN. MODE
0D55:	20 42 0F	JSR \$0F49	TO SLOW BLINK
0D5D:	0E	CTRL M ?	IN. MODE OFF?
0DBA:	4C 57 0F	JMP \$0F57	SLOWBLINK OFF
0DBD:	0C	CTRL L	START OF TEXT
SLOW 0F42:	A2 0A	LDX #0A	TURN ON
0F44:	BE 40 E1	STX #E140	SLOW
0F47:	A2 60	LDX #60	BLINKING
0F49:	BE 41 E1	STX #E141	CURSOR
0F4C:	20 1D F7	JSR \$F71D	GET A CHR.
0F4F:	C9 08	CMP #08	UNDER
0F51:	90 01	BCC #01	CTLH ?
0F53:	60	RTS	
0F54:	4C 60 0D	JMP \$0D60	
0F57:	20 BB 13	JSR #13BB	BLINK OFF
0F5A:	4C 09 0C	JMP \$0C09	WRCON +03
TXN 0F3A:	07	FROM HERE YOU HAVE 7	
0F3B:	60 20 73	CHR. TO TEXT THE MESSAGE	
0F3E:	20 66 69	'USE BLANKS TO SEPARATE	
0F41		FILES NAMES'	

Now I have been playing around with the 'samson' for some months and I think it's a great system because it's an open system that you can do almost anything with and see what you are doing with the monitorprogram. Now there are so much software that some standardisation would be in place.

Since last I have bought a printer (Epson LX80) and made some changes of the Wordprocessor to enable typing the printer codes directly on the screen while editing the text.

This can now be done in two ways. In the 'write insert mode' the cursor blink slowly and it is possible to write the chrs. \$00 to \$7E minus \$08 to \$0D that are used for cursor control, home, return and delete. This has the disadvantage that the ctrl.chrs. will not show on screen. Fortunately most printers can also use the high case chrs. f.ex. \$8B instead of \$1B for printer control. When ctrl. Q is pressed the cursor blink quickly and is small indicating that the next chr. will be added with 127. In this way it is also possible to use the printer grafik chrs..



BREAKTEST



## BASICODE

W. v. Dongen

In de Basicode van de NOS komt een subroutine voor om tijdens het lopende programma bepaalde input te verwerken, namelijk op regel 200.

Met een variatie op de joystick interface uit DE 6502 KENNER nr. 35 heb ik bijgaand programma gemaakt. Bij mij staat dit in Eprom op de aangegeven adressen. Het kan echter zonder een wijziging overal worden geplaatst.

Als joystick gebruik ik de goedkope van Philips, die ik zonder een aanpassing op de VIA (Versatile Interface Adaptor) heb aangesloten.

### Besturing joystick

17B1	A9 1F	BESTUR	LDAIM \$1F	BIT 0-1-2-3-4 VAN
17B3	8D 03 18		STA VADD	POORT A INGANG
17B6	A9 00		LDAIM \$00	GEEN BEVRIEZING
17B8	8D 0B 18		STA VACR	VAN INGANGSDATA
17BB	8D FF 07		STA GEBAS	GEHEUGENPLAATS LEEG MAKEN
17BE	AD 01 18		LDA VAD	HAAL STAND JOYSTICK OP
17C1	0A		ASLA	BIT 0-1-2-3-4
17C2	0A		ASLA	WORDT
17C3	0A		ASLA	BIT 3-4-5-6-7
17C4	29 F8		ANDIM \$F8	TEST BIT 3-4-5-6-7
17C6	F0 15		BEQ BESE	GEEN AKTIE?
17C8	29 F0		ANDIM \$F0	TEST BIT 4-5-6-7
17CA	F0 12		BEQ BESA	KNOP INGEDRUKT?
17CC	29 E0		ANDIM \$C0	TEST BIT 5-6-7
17CE	F0 14		BEQ BESB	AFWIJKING NAAR RECHTS?
17D0	29 C0		ANDIM \$C0	TEST BIT 6-7
17D2	F0 16		BEQ BESC	AFWIJKING NAAR BENEDEN?
17D4	29 80		ANDIM \$80	TEST BIT 7
17D6	F0 18		BEQ BESD	AFWIJKING NAAR LINKS?
17D8	A9 55		LDAIM \$55	NEE! DAN LETTER U(p)
17DA	8D FF 07		STA GEBAS	IN GEHEUGENPLAATS ZETTEN
17DD	60	BESE	RTS	
17DE	A9 46	BESA	LDAIM \$46	KNOP INGEDRUKT! LETTER F(ire)
17E0	8D FF 07		STA GEBAS	IN GEHEUGENPLAATS ZETTEN
17E3	60		RTS	
17E4	A9 52	BESB	LDAIM \$52	AFWIJKING RECHTS! LETTER R(ight)
17E6	8D FF 07		STA GEBAS	IN GEHEUGENPLAATS ZETTEN
17E9	60		RTS	
17EA	A9 44	BESC	LDAIM \$44	AFWIJKING NAAR BENEDEN! LETTER D(own)
17EC	8D FF 07		STA GEBAS	IN GEHEUGENPLAATS ZETTEN
17EF	60		RTS	
17F0	A9 4C	BESD	LDAIM \$4C	AFWIJKING NAAR LINKS! LETTER L(eft)
17F2	8D FF 07		STA GEBAS	IN GEHEUGENPLAATS ZETTEN
17F5	60		RTS	

### Basic-commando

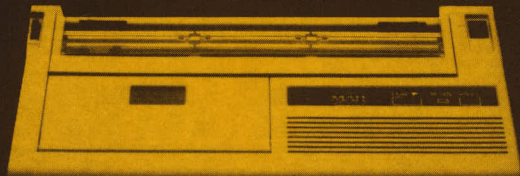
```
200 Q=USR("&"17B1",0):PO=PEEK(2047):REM GEHEUGENPLAATS GEBAS $07FF
201 IN$=CHR$(PO):RETURN
```



# Fabelachtig printen in kleur of zwart/wit



## OKIMATE 20



## OKI MICROLINE 182

Futuristische OKI-afdrukkers erkend de beste in Nederland. Nu ook voor iedereen betaalbaar geworden. Aansluitbaar op Commodore 64, BBC, MSX e.a. homecomputers.

OKImate 20 v.a. fl. 985.--  
Microline 182 v.a. fl. 1170.--  
excl. b.t.w. Bruto adviesprijs.

OKI printers, door computers voor computers.

Vraag de dealerlijst bij de officiële importeur

**Technitron B.V.**

Zwarteweg 110, Postbus 14, 1430 AA Aalsmeer

tel. 02977-22456 – telefax 02977-40968 – telex 13301